VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ
FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION

ÚSTAV TELEKOMUNIKACÍ
DEPARTMENT OF TELECOMMUNICATIONS

ROZPOZNÁVÁNÍ OSOB S POMOCÍ SNÍMKŮ OBLIČEJE
PEOPLE RECOGNITION USING FACIAL IMAGES

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE — Michal Lindovský
AUTHOR

VEDOUCÍ PRÁCE — Ing. Martin Rajnoha
SUPERVISOR

BRNO 2019

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ

Bakalářská práce

bakalářský studijní obor **Teleinformatika**
Ústav telekomunikací

Student: Michal Lindovský ***ID:*** 174343
Ročník: 3 ***Akademický rok:*** 2018/19

NÁZEV TÉMATU:

Rozpoznávání osob s pomocí snímků obličeje

POKYNY PRO VYPRACOVÁNÍ:

Popište nejpřesnější současné metody pro rozpoznávání osob. Teoreticky popište a porovnejte z hlediska přesnosti a výpočetního času projekty Facenet [1], konkrétně jeho implementaci OpenFace a Face Recognition [2]. Vytvořte webovou aplikaci pro rozpoznávání osob, která bude umožňovat rozpoznání osoby mezi několik milionů snímků v řádu několika sekund a také nahrávat a mazat nové osoby do databáze. Čas rozpoznání osoby snižte pomocí různého rozdělení množiny osob (pohlaví, věk...) a vytvořte algoritmus pro automatické zařazení do dané množiny.

DOPORUČENÁ LITERATURA:

[1] SCHROFF, Florian; KALENICHENKO, Dmitry; PHILBIN, James. Facenet: A unified embedding for face recognition and clustering. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 2015. p. 815-823.

[2] GEITGEY Adam. Face Recognition [online]. 2017 [cit. 2017-09-12]. Dostupné z: https://face-recognition.readthedocs.io/

Termín zadání: 1.2.2019 ***Termín odevzdání:*** 27.5.2019

Vedoucí práce: Ing. Martin Rajnoha
Konzultant:

prof. Ing. Jiří Mišurec, CSc.
předseda oborové rady

ABSTRAKT

Tato bakalářská práce se zaměřuje na rozpoznání osoby mezi několika miliony osob v řádech několika sekund. V rámci práce se porovnávají dva frameworky sloužící k rozpoznávání obličejů, OpenFace a Face Recognition. Jsou porovnávány výpočetní časy lokalizace a zakódování obličeje. Dále se porovnává přesnost rozpoznání v různých testech např. rozostřený obrázek, změna světlosti, věk osoby, použití slunečních brýlí. Vytvořená webová aplikace slouží k rozpoznání osoby v různých databázích osob. V aplikaci lze přidávat/odebírat databáze osob. Aplikace umožňuje do databáze zařadit osoby automaticky podle pohlaví nebo ručně. Pro zrychlení rozpoznání osoby lze využít více jáder procesoru.

KLÍČOVÁ SLOVA

Detekce obličeje, přesnost, rychlost, rozpoznání obličeje, konvoluční neuronové sítě, webová aplikace, rozpoznání pohlaví

ABSTRACT

This bachelor thesis focuses on the person recognition between several millions of people in a few seconds. As a part of my thesis is comparison of two programs which are used for recognizing faces - OpenFace and Face Recognition. Computing times of localization and face encoding are compared. The accuracy of recognition in various tests is compared as well, such as blurred image, brightness changes, age of person or usage of sunglasses. Created web application is made for recognizing people in different databases. Is possible to add or remove databases of people in the application. The application allows to subsume people into database by gender automatically or manually. Face recognition can be speeded up by using multiple processor cores

KEYWORDS

Face detection, accuracy, speed, face recognition, convolutional neural networks, web application, gender recognition

LINDOVSKÝ, Michal. *Rozpoznávání osob s pomocí snímků obličeje*. Brno, 2019, 56 s. Bakalářská práce. Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, Ústav telekomunikací. Vedoucí práce: Ing. Martin Rajnoha

PROHLÁŠENÍ

Prohlašuji, že svou bakalářskou práci na téma „Rozpoznávání osob s pomocí snímků obličeje“ jsem vypracoval samostatně pod vedením vedoucího bakalářské práce a s použitím odborné literatury a dalších informačních zdrojů, které jsou všechny citovány v práci a uvedeny v seznamu literatury na konci práce.

Jako autor uvedené bakalářské práce dále prohlašuji, že v souvislosti s vytvořením této bakalářské práce jsem neporušil autorská práva třetích osob, zejména jsem nezasáhl nedovoleným způsobem do cizích autorských práv osobnostních a/nebo majetkových a jsem si plně vědom následků porušení ustanovení § 11 a následujících autorského zákona č. 121/2000 Sb., o právu autorském, o právech souvisejících s právem autorským a o změně některých zákonů (autorský zákon), ve znění pozdějších předpisů, včetně možných trestněprávních důsledků vyplývajících z ustanovení části druhé, hlavy VI. díl 4 Trestního zákoníku č. 40/2009 Sb.

Brno ....

podpis autora

PODĚKOVÁNÍ

Rád bych poděkoval vedoucímu bakalářské práce panu Ing. Martinu Rajnohovi za odborné vedení, konzultace, trpělivost a podnětné návrhy k práci.

Brno ....

podpis autora

Obsah

Seznam obrázků

Seznam tabulek

Seznam výpisů

Úvod

V dnešní době je velice důležitá bezpečnost státu a ochrana lidí víc než kdy předtím. Pro zlepšení této bezpečnosti lze využít informační technologie např. k identifikaci osob. Jedním ze způsobů identifikace je rozpoznávání osob pomocí snímku obličeje. K tomu se obvykle využívají algoritmy strojového učení, spadající pod umělou inteligenci. To znamená, že lze naučit počítač detekovat obličeje osob a následně obličeje rozpoznat. Vychází to z podobného principu, jako když člověk rozpoznává nějakou osobu.

Dříve byl největší překážkou nedostatečný výpočetní výkon. Dnes, díky moderním technologiím, které zvýšily výpočetní výkon počítačů, lze takový program využít na kterémkoli počítači. Také se zlepšila přesnost rozpoznávání. Nastávají však další problémy, které nejsou plně vyřešeny. Například obličej krytý šálou nebo slunečními brýlemi je obtížné detekovat a následně rozpoznat konkrétní osobu. Další problém je natočení tváře v různém úhlu. Dnešní programy se částečně vypořádají s těmito problémy, ale stále nejsou schopny pracovat tak kvalitně jako lidský mozek.

Tato bakalářská práce se zabývá vytvořením webové aplikace pro rozpoznávání osob, která umožňuje přidávat osoby do databáze a také z databáze osoby odebírat, rovněž umožňuje vytvářet/rušit databáze.

Hlavním přínosem této práce je navržení webové aplikace na rozpoznávání osob, která je schopná rozpoznat určitou osobu mezi několika miliony osob za jednotky sekund. Taktéž musí webová aplikace umožnit rozdělit osoby do různých databází, aby rozpoznání netrvalo příliš dlouho a zmenšila se chyba rozpoznání. Zásadně je lepší rozdělit osoby podle různých kritérií (např. na muže a ženy).

Práce je strukturována následovně. První kapitola se věnuje teoretickému popisu rozpoznávání osob a popisuje stávající řešení. V druhé kapitole jsou popsány implementace OpenFace a Face Recognition, zároveň popisuje návrh webové aplikace pro rozpoznávání osob. Ve třetí kapitole jsou shrnuty výsledky této bakalářské práce.

1 Teoretická část

Lidé rozpoznávají osoby různě - například podle výšky, chůze, hlasu nebo také podle obličeje. Díky těmto znakům dokážeme přiřadit k hlasu nebo obličeji jméno osoby (jedinečný identifikátor). Problém však nastává, pokud chceme naučit rozpoznávat osoby stroj. Pro tento proces rozpoznávání je nejvhodnější rozpoznávání pomocí snímků obličejů.

Rozpoznávání osob je jedním typem biometrické technologie, kterou lze použít stejně jako otisky prstu pro identifikaci osob. Zatímco otisky prstů vyžadují aktivní účast osoby, rozpoznávání obličeje může být používáno bez spolupráce osob, tj. pasivně [5].

Mezi první průkopníky automatického rozpoznávání obličeje se řadí Woody Bledsoe, Helen Chan Wo a Charles Bisson. Během roku 1964 a 1965 společně pracovali na rozpoznávání lidské tváře za použití počítače. V roce 1966 práce pokračovala ve výzkumném ústavu na Stanfordově univerzitě, kde Peter Hart prováděl experimenty na databázi s více než 2000 fotografiemi. V roce 2006 byly testovány novější algoritmy pro rozpoznávání tváře a výsledky ukázaly, že nové algoritmy jsou desetkrát přesnější než algoritmy z roku 2002 a stokrát přesnější než v roce 1995, dokonce některé algoritmy byly schopny překonat při rozpoznávání tváři lidské účastníky a jednoznačně identifikovat totožná dvojčata [6].

1.1 K čemu může sloužit rozpoznávání osob

Rozpoznávání osob může sloužit takřka všude, kde je potřeba identifikovat neznámé osoby. Nebo také pokud chceme vyfiltrovat například různé fotografie s osobami.

V dnešní době se rozpoznávání osob pomoci snímku obličeje používá také v mobilních telefonech nebo počítačích. Rozpoznávání osob má velice širokou škálu použití. Může sloužit na veřejných místech, jako jsou letiště, vlakové/autobusové nádraží nebo v různých budovách. Jednoduše všude, kde se pohybuje velké množství lidí a je potřeba dbát na zvýšenou bezpečnost.

Například na letištích může rozpoznávání osob napomáhat policistům při bezpečnostní kontrole před odletem letadla nebo v případě příletu do země, kde se provádí celní kontrola. Takový systém lze využít na rozpoznání osob ve videích, které zachycují kamery rozmístěné v letištních halách nebo při osobní kontrole, kde si celní úředník vyfotografuje osobu a fotografií pošle k porovnání s databází.

Častým problémem v dnešní době jsou výtržnosti na sportovních zápasech respektive nejčastěji na fotbalových utkáních, kde neukáznění fanoušci demolují zařízení stadiónů nebo odpalují pyrotechniku a tím ohrožují jak sebe tak okolí. Proto je

prioritou takovým osobám zamezit další vstup nebo, v případě protiprávního jednání, provést identifikaci. V tomto případě pokud se použije systém na rozpoznávání osob, lze nepřizpůsobivým fanouškům znesnadnit či úplně zabránit vstupu na zápas.

Když bude vytvořena databáze potencionálně nebezpečných lidí, nebo bude-li napojena databáze celostátně hledaných nebo celosvětově hledaných osob na systém rozpoznávání osob, bude možné minimalizovat riziko vzniku teroristických útoků, protiprávních jednání nebo zvýšit úspěšnost dopadení osoby, na kterou je vydán zatykač.

Do této databáze mohou přidávat nebo odebírat záznamy (fotografie hledaných osob) bezpečnostní služby (PČR, BIS a další úřady k tomu určené). V případě, že se hledaná osoba objeví v místě, kde je systém nainstalován, objeví se obsluze systému informace o shodě s databází a informace bude zasílána konkrétní službě a následně se budou provádět kroky pro zadržení, sledování apod. Takové využití lze uplatnit na všech veřejných kamerách, jaké jsou například umístěny ve městech.

Rozpoznávání osob se může používat ve firmách pro hlídání, zda zaměstnanec má oprávnění vstupovat do místnosti, nebo na dané patro. Většinou takový přístup je ošetřen pomoci čipových karet, ale tuto kartu může zaměstnanec nepozorovaně odcizit kolegovi nebo zaměstnanec, který má přístup vezme svého kolegu do míst, kde mohou jen pověřené osoby.

Příklad využití ve firmách je následující, pokud se jedná o firmu se stovkami a víc zaměstnanců, tak ostraha nepozná na kamerovém záznamu, že osoba pohybující v daném místě, nemá do tohoto místa přístup. Pro zamezení takovému přístupu lze využít biometrické zabezpečovací systémy jako je autentizace pomocí otisků prstů nebo oční duhovky a přísnými pravidly vstupu. Pokud se však pohybuje v místě velké množství osob a není potřeba mít vysoké zabezpečení pomocí biometriky, tak lze využít rozpoznávání osob pomocí snímků obličeje. Systém pro rozpoznávání osob alespoň nezdržuje oprávněné osoby při vstupu, ale naopak bezpečností službě v objektu usnadní práci s identifikaci neoprávněného vstupu.

1.2 Popis problematiky

Problematika rozpoznávání osob s pomoci snímků obličeje v dnešní době není stále vyřešena. Avšak přesnost vědeckých prací v některých kritériích začala překonávat lidi [4].

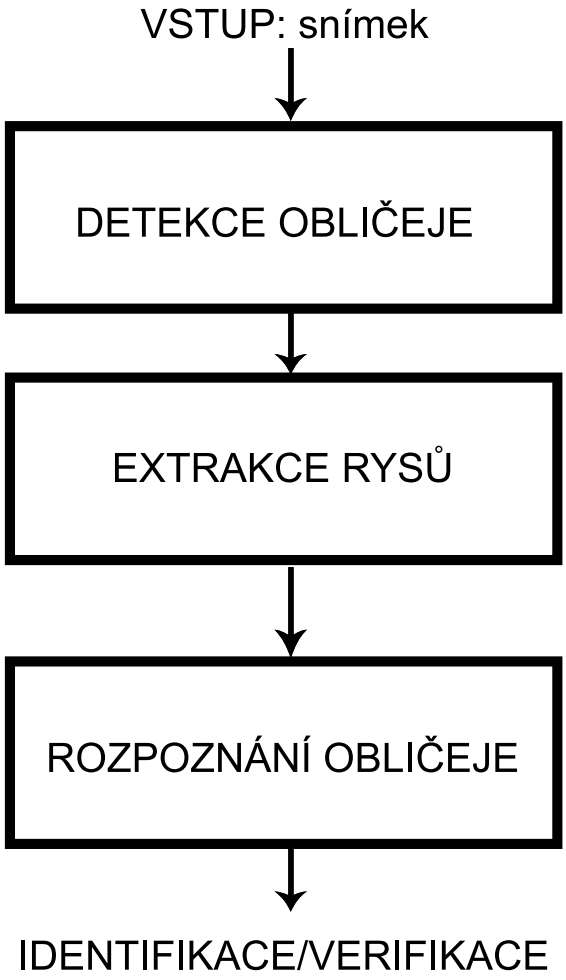
Rozpoznávání osob pomoci snímků obličeje je proces, při kterém je k osobě přiřazen jedinečný identifikátor. Tento proces lze rozdělit na: detekci obličeje, detekci bodů obličeje (extrakce rysu obličeje) a rozpoznávání obličeje [1].

Tyto úkony dělá lidský mozek zcela automaticky bez jakéhokoliv podnětu. Díky dnešní moderní výpočetní technice lze tento proces aplikovat za pomoci umělé in-

teligence do strojového vidění, protože dnešní počítače mají mnohonásobně vyšší výpočetní výkon než počítače z minulého století. Tyto procesy dnes trvají desítky až stovky milisekund, výjimečně sekundy. Doba zpracování záleží na použitém výpočetním výkonu a jaká vstupní data se zpracovávají [1].

Obecný systém rozpoznávání obsahuje tři základní kroky [1]:

- **Detekce obličeje:** rysy (oči, nos, ústa) se častou používají při detekci obličeje.
- **Extrakce rysů:** může probíhat současně s detekcí obličeje.
- **Rozpoznávání obličeje:** V posledních 30 letech bylo navrženo mnoho metod pro rozpoznávání obličeje. Jedná se o náročný proces, ale přesto tento problém přilákal vědce. Navrhli různé principy: psychologie, rozpoznávání vzorů, počítačové vidění a počítačová grafika, neuronové sítě.



Obr. 1.1: Obecný systém rozpoznávání obličeje [1].

Aby byla jasná kategorizace na vysoké úrovni, postupuje se podle pokynů navržených psychologickou studií o tom, jak lidé rozpoznávají osoby [7]:

- **Holistické metody shody:** Tyto metody využívají celou plochu tváře jako surový vstup do systému rozpoznávání. Jedním z nejlepších příkladu holistické metody je Eigenface[1].
- **Strukturální metody shody:** Obvykle jsou v těchto metodách poprvé extrahovány lokální rysy, jako jsou například oči, nos a ústa. Jejich umístěni a lokální statistiky (geometrické nebo vzhledové) jsou převáděny do strukturního klasifikátoru.
- **Hybridní metody:** Používají kombinaci holistickým a strukturálních metod. Obecně se v hybridních metodách využívají 3D obrázky. Obraz tváře je zachycen ve 3D, což systému umožnuje určit tvar brady nebo čela.

1.3 Problémy současnosti

Při rozpoznávání osob pomocí snímků obličeje nastává mnoho problémů. Před dvaceti lety byl hlavním problémem nedostatečný výpočetní výkon, který se dnes eliminoval novou moderní technikou. Mezi problémy se řadí například při detekci obličeje barva kůže, ale hlavní problém při detekci obličeje nastává, když je obličej příliš zakrytý slunečními brýlemi nebo šálou. Jestliže není detekován obličej, tak není možné přistoupit k rozpoznávání.

Moderní obličejové detektory správně detekují obličeje, které jsou vhodně natočené. Výzkumy se taktéž zaměřují na problémy s přehnanými výrazy obličeje (mimika) a extrémním osvětlením, které mohou vést k vizuálním změnám vzhledu obličeje. Příliš tmavá scéna s tmavým obličejem může snížit úspěšnost detekování [9]

V roce 2002 proběhl test FRVT[2] 2002, kterého se zúčastnil systém Identix, Eyematic a další. Test obsahoval přes 121 tisíc snímků a 37 tisíc osob. Jednalo se o snímky z vízové databáze amerického ministerstva zahraničí, které byly shromážděny v Mexiku, tzn. že fotografie byly v dobré kvalitě s dobrým osvětlením. Z testu vyplynulo, že i věk (snímky tváří pořízených v různém časovém období) hraje v rozpoznávání svou roli [10].

Jedná se o zásadní problém, protože pořízení snímku v dlouhém časovém rozestupu ovlivňuje přesnost rozpoznání osoby zejména v rámci velkého množství dat. Stárnutí lidského těla je přirozený proces, který nelze zastavit. Během dětství jsou změny obličeje způsobeny v důsledku vývoje tvaru hlavy, v pozdějších stadiích života dochází k další změnám, které ovlivňuji například vzhled kůže a její pružnost.

[1] Eigenface - algoritmus pro detekování a rozpoznání obličeje představený v roce 1991 [8].

[2] Face Recognition Vendor Test [10]

Kromě anatomických faktorů také ovlivňuje vzhled obličeje životní styl (kouření, alkohol, drogy, jídlo, stres) [11].

1.4 Základní metody rozpoznávání

K vyhledávání a rozpoznávání obličejů slouží mnoho postupů. Například pro nalezení obličeje lze použít HOG[3] detektor, ale v posledních letech se častěji začíná využívat strojového učení. Natrénované konvoluční neuronové sítě se dnes využívají k rozpoznání různých objektů, například dopravních značek nebo registračních značek vozidel [12][13].

1.4.1 Orientační body

Orientační body (v angličtině landmarks) jsou důležité geometrické rysy na tvářích. Orientační body lze klasifikovat jako [14]:

- **Anatomické orientační** body jsou přísně definované biologicky významné body, které byly definovány odborníky. Jsou to body, které my jako lidé vnímáme (oči, nos apod.).
- **Matematické orientační** body jsou definovány podle některých matematických nebo geometrických vlastností.
- **Pseudo-orientační** body jsou definovány kolem obrysu objektu nebo mezi dvěma existujícími anatomickými nebo matematickými orientačními body.

Pokud lidi rozpoznávají obličej, tak k tomu využívají anatomické orientační body. To znamená, že detekuji například nos a jeho velikost, stejně tak oči nebo ústa. Jako lidi to provádíme zcela automaticky, bez jakéhokoliv podnětu. Naopak matematické orientační body jsou důležité pro automatické rozpoznávání obličeje, protože tyto body mohou být popsány matematickými funkcemi. To znamená, že s matematickými body pracují například stroje [14].

Vzhledem k tomu, že orientační body jsou mapovány na dvourozměrných snímcích obličeje, je vhodné používat pro každý orientační bod číselný zápis v komplexním čísle. Obličej f může být modelován pomocí N dimenzionálního vektoru orientačních bodů [14]

$$f = [l_1, l_2, ...l_N]^T \tag{1.1}$$

kde

$$l_i = x_i + \sqrt{-1}y_i, (x_i, y_i) \tag{1.2}$$

[3]Histogram of Oriented Gradients - detektor založený gradientové orientaci v lokalizovaném obraze [12].

jsou souřadnice orientačního bodu l_i.

Dobře definované orientační body mohou být použity pro rozpoznávání obličeje po Euklidovské transformaci. Nechť souřadnice orientačních bodů jsou (x_{1i}, y_{1i}) a (x_{2i}, y_{2i}) budou konfiguraci orientačních bodů $f_1 = [l_{11}, \ldots, l_{1N}]^T$ a $f_2 = [l_{21}, \ldots, l_{2N}]^T$.

Euklidovská transformace je definovaná jako [14]

$$\begin{bmatrix} x_{1i} \\ y_{1i} \end{bmatrix} = \begin{bmatrix} a \\ b \end{bmatrix} + \beta \cdot \begin{bmatrix} \cos(\theta) & -\sin(\theta) \\ \sin(\theta) & \cos(\theta) \end{bmatrix} \cdot \begin{bmatrix} x_{2i} \\ y_{2i} \end{bmatrix} \tag{1.3}$$

kde a a b reprezentuje překlad, β je kladné reálné číslo odpovídající měřítku, $\theta \in [0, 2\pi]$ odpovídá rotaci [14].

1.4.2 Neuronové sítě

Neuronové sítě jsou inspirovány biologickým mozkem. Umělý neuron (tzv. perceptron) vychází z funkce biologického neuronu. Neuron je základní funkční jednotka nervové soustavy, který je ve velkém množství obsažen ve všech mozcích živočichů, zajišťuje vedení vzruchů (elektrických signálu). K vedení elektrických signálu slouží vzrušivá membrána, která vzruchy v synapsích (tj. spojení dvou neuronů) posílá dál [15].

Algoritmus zpětné propagace

Algoritmus zpětné propagace udává, jak by se měly měnit vnitřní parametry, které se používají k výpočtu reprezentace v každé vrstvě z reprezentace z předchozí vrstvy. Algoritmus se využívá v přibližně 80% všech aplikacích neuronových sítí, jedná se o zpětné šíření chyby [15].

První vrstva (vstupní) obsahuje surová data. Za první vrstvou, tzv. vstupní je skryto n vrstev a jako poslední vrstva je výstupní, která reprezentuje výstup celé neuronové sítě. Algoritmus obsahuje tři etapy [15]:

- dopředné šíření vstupního signálu trénovacího vzoru
- zpětné šíření chyby
- aktualizace váhových hodnot na spojeních

Při učení je první krok dopředné šíření signálu, každý neuron vstupní vrstvy obdrží signál, který zprostředkuje ostatním neuronům skryté vrstvy. Následně neurony vypočítávají svou aktivační funkci a předávají svůj výstup další vrstvě, takto se hodnoty dostanou až do výstupní vrstvy. Obdobný postup funguje u biologického systému, kde vstupem může být například obraz, který je zachycený pomocí tyčinek v oku [15].

Hluboké učení

Konvekční techniky strojového učení měly jen omezenou schopnost zpracovávat přirozené data v surové formě. Hluboké učení je jedna z technik strojového učení, kterou lze využít k identifikaci objektů v obraze, k přepisu řeči do textu, k výběru relevantních výsledků vyhledávání atd. Umožňuje výpočtovým modelům, které se skládají z více vrstev zpracování, naučit se reprezentaci dat s více úrovněmi abstrakce [16].

Metody hlubokého učení jsou metody reprezentace dat s více úrovněmi reprezentace, získané kompozici nelineárních modulů, z níž každá úroveň transformuje reprezentaci. V hlubokém učení se využívá algoritmus zpětné propagace. [16].

Konvoluční neuronové sítě

Konvoluční neuronové sítě (anglicky Convolutional neural network nebo také CNN[4]) jsou speciálním druhem. U konvoluční neuronové sítě je předpoklad, že vstupní data budou například obrázky, audio apod. Stejně jako většina neuronových sítí jsou CNN trénované pomoci algoritmu zpětné propagace. CNN používají variaci vícevrstvých perceptronů navržených tak, aby rozpoznaly vizuální vzory například z pixelových snímků [16][17].

Vrstvy CNN se obvykle skládají z těch třech typů [18]:

- konvoluční vrstva
- sdružovací vrstva
- plně propojená vrstva

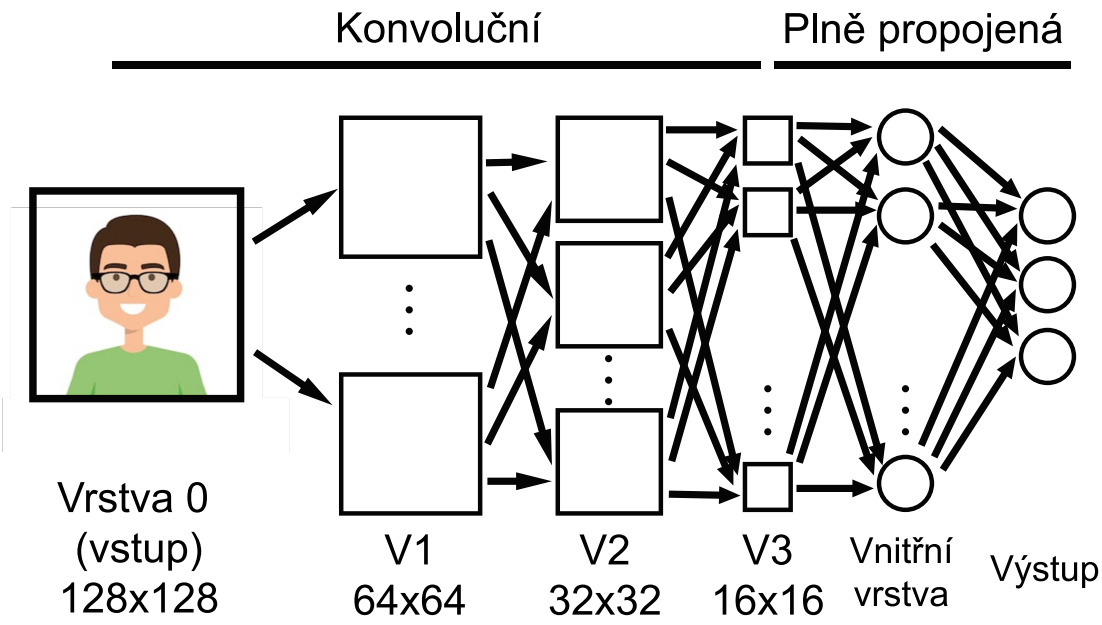
Každá vrstva používá různé filtry, obvykle to jsou stovky nebo tisíce. Během fáze trénovaní CNN se automaticky naučí hodnoty na základě úkolů, které chceme provést. Například v klasifikaci obrázků se CNN může naučit rozpoznávat okraje ze surových pixelů v první vrstvě, pak pomocí okrajů rozpoznat jednoduché tvary v druhé vrstvě a potom využít tyto tvary ve vyšších úrovních. Poslední výstupní vrstva reprezentuje klasifikaci do kategorií. [18][3].

Konvoluční vrstva: Jedná se o strukturu, která je velmi důležitá. Ve struktuře jsou neurony rozloženy uvnitř každé vrstvy. Vytvářejí spoustu aktivačních map[5]. Každý neuron má pouze lokální spojení, které tvoří jeho receptivní pole předchozí vrstvy. V rámci jedná aktivační mapy sdílejí neurony stejnou váhu. [18].

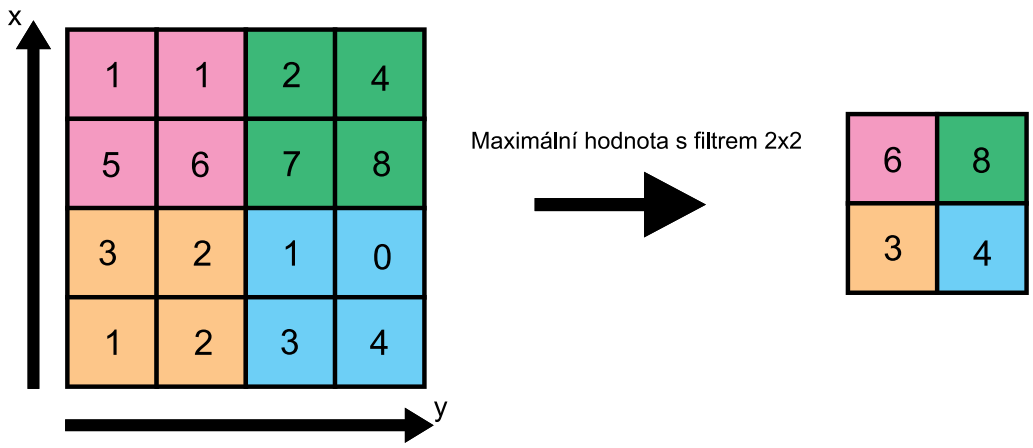
Sdružovací vrstva: Klíčové jsou sdružovací vrstvy, které se obvykle používají po konvolučních vrstvách. Jedná se o sdružování vrstev dílčího vzorku z jeho vstupu. Nejběžnější způsob, je použití metody tzv. max-poolingu (maximální hodnota v oblasti), zobrazeno na obrázku 1.3 [18].

[4]Convolutional neural network

[5]Aktivační mapa - obsahuje mapu vlastností vytvořených konvoluční vrstvou [3].



Obr. 1.2: Ukázka architektury konvoluční neuronové sítě [2].



Obr. 1.3: Znázornění výběru nejvyšší hodnoty v oblasti [3].

Plně propojená vrstva: Neurony v této vrstvě mají plné spojení se všemi aktivacemi v předchozí vrstvě, jak je patrné z běžných neuronových sítí. Výstupem plně propojené vrstvy je rozřazení do tříd.[18].

1.4.3 Stávající řešení rozpoznávání osob

Existují projekty, které jsou veřejně dostupné. Pokud to umožňuje licence, lze veřejně dostupné programy na rozpoznávání osob volně používat. Také existují neveřejné programy, kterým je například projekt DeepFace od společnosti Facebook Inc. [19]

DeepFace

Za implementaci DeepFace stojí společnost Facebook Inc., která svůj projekt přestavila v roce 2014. Jedná se konvoluční neuronovou síť, která byla trénovaná na datasetu o velikosti asi čtyři tisíce osob. Každá osoba měla 800 a 1200 snímků obličeje a posledních 5% snímků bylo vynecháno pro testování [19].

DeepFace obsahuje 120 miliónu parametrů a je trénován na 3D zarovnaném RGB obrázku o velikosti 152x152 pixelů. Na datové sadě LFW[6] má úspešnost 97,35% [19] [20].

FaceNET

Florian Schroff, Dmitry Kalenichenko, James Philbin představili algoritmus v roce 2015. Úspěšnost rozpoznávání obličejů je 99,63% na datové sadě LFW a na YouTube Faces DB je úspěšnost 95,12%[21][20][22].

Algoritmus FaceNET pracuje s konvoluční neuronovou sítí. Tato sít byla trénována na neveřejné databázi obličejů od společnosti Google Inc., vstupní velikosti každého obrázku byla 220 x 220 pixelů. Výstupem konvoluční neuronové sítě je 128 dimenzionální (128D) vektor [21].

Z dokumentu, který je veřejně dostupný vychází implementace OpenFace a další jiné implementace.

OpenFace

Jedná se o implementaci, která je založena na myšlence popsané dokumentem FaceNET. Rozpoznávání obličeje probíhá za pomocí knihovny Dlib[7] a hlubokých neuronových sítí. Pro trénovaní neuronové sítě se používá framework Torch[8], díky kterému lze natrénovat vlastní neuronovou síť na svých datech [21][4][23].

OpenFace je trénován na veřejných datových sadách jako je LFW, FaceScrub[9] a CASIA-WebFace, které obsahují více něž 500.000 obrázků obličejů, na datové sadě LFW autoři uvádějí úspěšnost 92,92% [4][24].

[6]Labeled Faces in the Wild - dataset obsahující 5749 lidí s více než 13000 obrázky tváří [20].

[7]Knihovna obsahující algoritmy a nástroje pro strojové učení. `http://dlib.net/` [23].

[8]Framework podporující strojové učení `http://torch.ch/`.

[9]Dataset obsahující přes 100000 obličejů `http://vintage.winklerbros.net/facescrub.html`.

Face Recognition

Implementace vychází z OpenFace a dokumentu FaceNET. Autor vytvořil program kvůli nepřehlednosti a složitosti OpenFace. Face Recognition funguje na stejném principu jako OpenFace, ale neumožňuje natrénovat vlastní konvoluční neuronovou síť. K rozpoznávání obličeje se využívá pouze knihovna dlib, která od února 2017 obsahuje natrénovanou konvoluční neuronovou síť. Dlib má 99,38% úspěšnost rozpoznání tváře na LFW databázi. První funkční verze byla zveřejněna 13.3.2017 [21][25]

1.4.4 Teoretické zhodnocení frameworku

Podle zveřejněných dokumentů s výsledky testů je nejlepším programem FaceNET, který dosahuje 99,63% úspěšnosti na datové sadě LFW. Ovšem tyto zveřejněné údaje nelze nijak ověřit, jelikož se jedná soukromý program [21].

Následuje Face Recognition s úspěšnosti 99,38%, ale není to způsobeno programem samotným, nýbrž použitou knihovnou Dlib. Samotný Face Recognition pouze ulehčuje práci s knihovnou Dlib [25].

Třetím programem podle úspěšnosti 97,35% je DeepFace, ovšem tento výsledek úspěšnosti byl v roce 2014 a nyní může být lepší. Do dnešní doby mohla společnost Facebook Inc. zapracovat na vylepšení a posunout úspešnost na stejnou úroveň jako FaceNET, protože Facebook má přístup k velkému objemu obrazových dat [19].

Podle teoretického srovnání úspěšnosti nejhůře dopadl OpenFace s úspěšností pouze 92,92%, ale jeho výhodou je, že si lze natrénovat vlastní konvoluční neuronovou sít na vlastních datech a to přináší obrovskou výhodu oproti Face Recognition.

2 Praktická část

2.1 Praktické porovnání frameworků

Jedním z cílů této bakalářské práce bylo porovnání dvou frameworků, tj. OpenFace a Face Recognition a následně vytvořit webovou aplikaci, která umožní rozpoznávat osoby podle snímku obličejů. Nejprve byly nastudovány dostupné informace k frameworkům, pak byly prohlídnuty kódy frameworkům, aby se pochopila funkce frameworků OpenFace a Face Recognition. Funkce obou frameworků je popsaná níže.

Po nastudování a vyzkoušení funkcí frameworků se začalo provádět samotné testování měření časů a přesností. Výsledky měření jsou popsané v kapitole 3.

2.1.1 OpenFace

Při rozpoznávání používá klasifikaci dat. Tj. pokud máme více kódování obličejů jedné osoby (více obrázků tváře jedné osoby), provádí se vytvoření jednoduchého klasifikátoru [4]

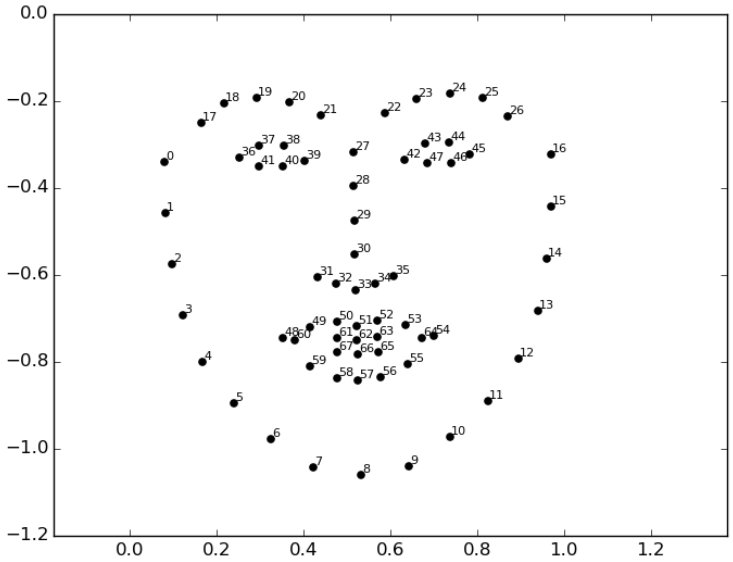
K vyrovnání tváře (pokud je obličej nějak nakloněný) se používá afinní transformace souřadnic (*cv2.getAffineTransform*), ale u OpenFace je zarovnání omezeno pouze na změnu velikosti, rotaci a překlad. Třída, která to umožňuje, obsahuje funkci align, která zarovná a změní velikost obličeje (například zmenší na 96x96 pixelů), snímek transformuje tak, aby se na každém snímku objevily orientační body (například oči nebo nos). Snímek zarovnává podle vzoru na obrázku 2.1 [4]

Zarovnaný obličej následně vstupuje do neuronové sítě. Výstupem neuronové sítě je 128D vektor. Při porovnávaní snímků se pouze porovnávají samotné vektory (jedná se pouze o matematickou operaci). To znamená, že nejdéle trvá nalezení obličeje a výpočet vektorů pomocí neuronové sítě.

K zarovnání snímku před vstupem do neuronové sítě slouží třída *AlignDlib*. Součástí třídy je funkce *align*, která obsahuje tyto vstupní parametry:

imgDim (int) – Délka hrany v pixelech, na které je velikost obrázku změněna.
rgbImg (numpy.ndarray) – RGB snímek pro zpracování.
bb (dlib.rectangle) – Rámeček, který ohraničuje obličej k zarovnání.
Landmarks (list (x, y)) – Nalezené orientační body. Orientační body nalezeny na bb, pokud nejsou k dispozici.
landmarkIndices (list) – Indexy ke tranformaci.
skipMulti (bool) – Přeskočí obrázek, pokud bylo zjištěno více obličejů.

Funkce vrací zarovnaný RGB obrázek. Návratová hodnota je numpy.ndarray [4].



Obr. 2.1: 68 obličejových bodů [4].

Výpis 2.1 je ukázka kódu, jak může vypadat nahraní a následné zpracování obrázků. Nejdříve se načte obrázek pomoci knihovny OpenCV[1], který se dále zpracovává. Zpracování probíhá následovně: provede vyznačení místa, kde se obličej nachází. Snímek se zpracuje pomoci třídy *align*, v posledním kroku zpracovány snímek prochází neuronovou sítí. Výstupem je vektor o velikosti 128 [4].

Výpis 2.1: Vytvořeni vektoru obličeje.

```
1  def getRep(imgPath):
2      bgrImg = cv2.imread(imgPath)
3      if bgrImg is None:
4          raise Exception("Unable to load image: {}".format
       (imgPath))
5      rgbImg = cv2.cvtColor(bgrImg, cv2.COLOR_BGR2RGB)
6
7      bb = align.getLargestFaceBoundingBox(rgbImg)
8      if bb is None:
9          raise Exception("Unable to find a face: {}".
       format(imgPath))
10
```

[1]Knihovna pro manipulaci s obrazem.

```
11      alignedFace = align.align(args.imgDim, rgbImg, bb,
    landmarkIndices=openface.AlignDlib.OUTER_EYES_AND_NOSE
    )
12      if alignedFace is None:
13          raise Exception("Unable to align image: {}".
    format(imgPath))
14
15      rep = net.forward(alignedFace)
16  return rep
```

Porovnání různých obličejů

Pro zjištění, zda se dva porovnávané obličeje shodují, je potřeba vypočítat vzdálenost vektorů. Pro výpočet se využívá funkce *numpy.dot*. Výsledkem je pak desetinné číslo v rozsahu od 0 do 4, které říká, jak moc si jsou obličeje podobné. Autor uvádí, že obličeje s hodnotou menší než 0,99 lze považovat za shodné [4].

2.1.2 Face Recognition

Face Recognition funguje stejně jako OpenFace, ovšem nemá vlastní neuronovou síť, nýbrž využívá neuronovou síť, která je součásti knihovny Dlib. Tato neuronová síť slouží k vytvoření 128D[2] vektoru obličeje [23].

K rozpoznávání obličeje slouží 128D (dimenzionální) vektor. To znamená, že kódování obličeje je popsané pouze řadou 128 čísel. Díky použití 128D vektoru, lze tyto čísla například ukládat do CSV souboru pro pozdější práci nebo zálohu [25].

Stejně jako u OpenFace se počítá vzdálenost mezi vektory pro určení podobnosti obličejů, v tomto případě se vypočítává Euklidovská vzdálenost. Podle autorů lze za shodu považovat hodnoty menší než 0,6. K uložení řady čísel využívá NumPy pole, jedná se o knihovnou vhodnou pro vědecké výpočty v Pythonu. Tato knihovna obsahuje N dimenzionální pole. NumPy je vedený pod licencí BSD[3] [25][27]

Ke zpracování obličeje používá knihovnu dlib, která před samotným vstupem do neuronové sítě provádí zarovnání obličeje. Dlib používá k zarovnání 68 obličejových bodů. Tyto body se používají k vypočtení rozdílu mezi orientačními body vstupní tváře a body v průměrném modelu. Na základě těchto bodových rozdílů provádí transformaci [25]:

- Natočení obličeje, aby byl obličej vzpřímený.
- Vycentrování obličeje, aby byl obličej ve středu.

[2] 128 dimenzionální vektor

[3] Berkeley Software Distrobution - Umožňuje volné šíření licencovaného obsahu [26]

- Změní rozměr na 150x150 pixelů s pevným vnitřním okrajem (25

Protože Face Recognition používá neuronovou síť z knihovny Dlib, tak neumožňuje vytrénování vlastní sítě. Autor program vytvořil, aby zjednodušil složitost programu OpenFace.

Popis funkcí

***load_image_file()*:** nahraje obrázek do paměti, funkce obsahuje tyto parametry:

file - cesta, kde se obrázek nachází

mode - formát pro převod obrazu

***face_locations()*:** lokalizace obličeje, vrací XY souřadnice místa, kde je tvář umístěna. Jedná se jen o čtyři souřadnice ohraničení kolem obličeje: horní, dolní, levá a pravá. Parametry funkce jsou:

img – obrázek v poli *numpy.ndarray*

number_of_times_to_upsample – kolikrát se má prohledat obrázek. Při vysokém čísle lze najít i malé obličeje

model – který model detekce obličeje se má vyžít

***face_encodings()*:** za pomoci neuronové sítě vytvoří 128D vektor, který se následně používá k rozpoznávání osob. Parametry:

face_image – snímek, který obsahuje jednu nebo více tváří

known_face_locations – ohraničující rámečky každé tváře, pokud je již známe.

num_jitters – kolikrát se bude opakovat výpočet kódování.

***face_distance()*:** výpočet euklidovské vzdálenosti

- face_encodings – List zakódovaných obličejů k porovnání
- face_to_compare – zakódovaný obličej, který se má porovnávat

2.2 Implementace řešení

Úkolem této bakalářské práce je vytvoření webové aplikace pro rozpoznávání osob, která umožní rozpoznávání osoby mezi několika miliony snímků v řádech jednotek sekund. Dalším požadavkem je přidávání nebo odebírání osob z databáze. Webová aplikace je implementováná v jazyce Python (verze 3.6).

Na základě praktických měření a testů psaných v kapitole 3 byl pro implementaci webové aplikace vybrán framework Face Recognition, kvůli své vysoké přesnosti oproti frameworku OpenFace.

2.2.1 Návrh aplikace

Nejdůležitější v celé aplikaci je způsob uchování vektorů obličejů a k nim mít vhodně přiřazenou identifikaci (ID obličeje nebo jméno). Protože je velice důležitá rychlost rozpoznaní, tak se hned na začátku vyloučila možnost ukládání dat do různých SQL databází. Proto je potřeba všechny data mít uložené v RAM paměti, ať má CPU přímý přístup k těmto datům.

Seznam, který se načítá do RAM paměti a obsahuje jména osob s vektory jejich obličejů, je v této práci práci označen jako databáze[4]. K pojmenování různých databází se používá v Pythonu slovník. Takovýmto způsobem lze uložit statisíce až miliony záznamu do RAM paměti, záleží na velikosti používané RAM. Pro 1 milion záznamu je potřeba přibližně 1,3 – 1,5 GB paměti.

Složení slovníku (v aplikaci pojmenovaný jako *globalList*) je následující:

Výpis 2.2: Vzor databází.

```
1 globalList = {
2 'default' : [[id_0, id_1, ...],[[vektor_0], [vektor_1],
    ...]]
3 'databaze_0' : [[id_0, id_1, ...],[[vektor_0], [vektor_1
    ], ...]]
4 }
```

Ukázalo se, že se jedná o nejvhodnější řešení. Všechny databáze jsou na jednom místě, ale v případě výpadků elektrického proudu se všechny data ztratí, proto je potřeba data ukládat na energetický nezávislý disk. Ukládat se bude každá databáze samostatně, nikoliv celý *globalList*, lze takto archivovat jen zvolené databáze.

Snížení času rozpoznávání

Výpočet Euklidovské vzdálenosti mezi dvěma vektory trvá v řádech mikrosekund, ale s každým dalším vektorem tento čas narůstá. Při jednom miliónu snímků to může trvat jednotky sekund. Aby vyhodnocení netrvalo příliš dlouho je prioritou čas snížit na minimum.

Pro snížení času byly navrhnuty dva způsoby. Prvním možným způsobem je rozdělit jednu databázi osob na dvě či více databází (například na databázi mužů a žen). Druhý způsob je použití více procesorů na porovnávání vektoru neznáme osoby s vektory v databázi (tím pádem je potřeba databázi při použití více procesorů rozdělit na několik částí).

[4]databáze = seznam o velikosti 2. Index 0 obsahuje seznam jmen a index 1 obsahuje pole s vektory.

Rozpoznáním pohlaví lze rozdělit osoby na dvě databáze, tj. databáze mužů a databáze žen. Do aplikace byl implementován již fungující model rozpoznání pohlaví, protože pro natrénování vlastního modelu nebyl k dispozici dostatečně velký dataset mužů a žen.

Jedná se o model s MIT[5] licencí vytvořený pro použití v reálném čase, který rozpoznává pohlaví. V tomto příkladu užití se model nebude využívat v reálném čase. Vstupním parametrem je snímek, který je zarovnaný a jeho velikost je 48x48 pixelů. Autoři uvádějí přesnost rozpoznání pohlaví 96% na databázi IMDb-Face [28][29].

V případě více procesorů byl navržený algoritmus, který má zajistit rychlejší výpočet Euklidovské vzdálenosti mezi vektory a rychlejší seřazení výsledků. To by mělo zapříčinit snížení času rozpoznání. Postup je následující:

1. Rozdělení databáze na menší části – seznam jmen (či ID) a seznam vektorů se rozdělí podle počtu zvolených procesů (jeden proces odpovídá jednomu jádru v CPU). V této bakalářské práci jsou použité 4 procesy, tzn. že se databáze rozdělí na ¼.
2. Proces má určenou část databáze – po spuštění procesů si každý proces načte pouze danou část databáze a s ní dále pracuje.
3. Procesy rozpoznávání běží nezávisle – výpočet Euklidovské vzdálenosti a následné seřazení výsledku probíhá nezávislé na ostatních procesech.
4. Uložení výsledků pro aplikaci – po zpracování každý proces uloží svůj výsledek do zvláštních souborů pro aplikaci, tyto soubory si aplikace načte, zpracuje a zobrazí výsledek.

2.2.2 Výběr webového frameworku

V programovacím jazyku Python lze vybírat z několika webových frameworků. Nejznámějším a nejrozšířenějším webovým frameworkem je Django[6], dále to jsou Flask[7], CherryPy[8], Pyramid[9] apod.

Flask

Pro webovou aplikaci byl zvolený webový framework Flask, který plně vyhovuje požadavkům na vytvoření webové aplikace. Jedná se o odlehčený framework, který je velice flexibilní. Hlavním požadavkem byla rychlá implementace (základní použití je velice jednoduché oproti frameworku Django).

[5] MIT – svobodná licence [26].
[6] `https://www.djangoproject.com/`
[7] `http://flask.pocoo.org/`
[8] `https://cherrypy.org/`
[9] `https://trypyramid.com/`

Pro použití je nutno stáhnout balíček (např. v příkazovém řádků použít příkaz *pip install Flask*). Pro samotné první použití stačí naimportovat třídu Flask a vytvořit instanci.

2.2.3 Další důležité moduly

Tato část popisuje potřebné knihovny a pomocné moduly, které jsou potřeba pro správnou funkci webové aplikace.

Dlib

Pokud má Face Recognition pro detekci obličejů používat GPU[10] je potřeba knihovnu Dlib zkompilovat pro grafickou kartu (předpokladem je mít tuto kartu v zařízení), v opačném případě není potřeba kompilovat Dlib a stačí opět pomocí příkazové řádky stáhnout knihovnu Dlib. Protože z předchozích testů vyplynulo, že při použití grafické karty pro detekci obličejů byl čas detekce delší než při použití HOG detektoru (který běží na CPU[11]), byl pro detekci obličeje zvolen HOG detektor.

Numpy

Knihovna, která poskytuje podporu pro práce s maticemi, vektory nebo vícerozměrnými poli. Do Numpy pole se ukládají vektory, rovněž v Numpy poli. [27]

Multiprocessing

Modul *multiprocessing* slouží pro vytváření nových procesů nebo vláken a následnou jejich správu [30].

V aplikaci slouží modul pro urychlení rozpoznávání tváře a seřazení výsledků. Aplikace je navrhnuta tak, že se seznam osob rozdělí podle počtu zvolených procesorů (jeden spuštěný proces běží na jednom jádru nezávisle na ostatních).

Pickle

Modul *pickle* implementuje binární protokol pro serializaci a deserializaci objektů v Pythonu. [30]

Pomoci tohoto modulu se ukládají na disk nebo z disku nahrávají všechny databáze. Pickle se taktéž používá při více procesorovém rozpoznávání. Za pomoci souborů uložených v binární podobě spolu komunikuje hlavní aplikace s běžícími procesy sloužící k výpočtu vzdálenosti.

[10] Grafická procesorová jednotka

[11] Centrální procesorová jednotka

2.2.4 Funkce třídy FRapp

Třída *FRapp()* byla vytvořena, aby obsahovala všechny důležité funkce pro správný chod celé webové aplikace. Tuto třídu lze implementovat i v jiných aplikacích např. do konzolové nebo desktopové aplikace.

V této podkapitole jsou popsané základní nejdůležitější funkce celé aplikace. Pro přehlednost v kódu jsou všechny funkce rozděleny do několika souborů a tyto soubory jsou uloženy ve složce classes. Tyto funkce se pak importují v souboru s názvem *fr.py* a odsud se volají jejich funkce. Jedná se o soubory:

- fr_add.py
- fr_append.py
- fr_delete.py
- fr_encoding.py
- fr_gender.py
- fr_load.py
- fr_recognition.py
- fr_save.py
- fr_settings.py
- fr_support.py
- fr_upload.py

loadGlobalList()

Jedna z funkcí, která volá při spuštění aplikace. Tato funkce slouží pro nahrání databází z uložených dat (uloženými daty se myslí jeden seznam obsahující seznam jmen a Numpy pole obsahující 128D vektory). V případě, že neexistují žádné uložené data, tak se provede vytvoření prázdných databází podle uloženého nastavení v souboru *configlist.ini*.

startMultiprocessing()

Spouští zvolený počet procesů (fyzických jáder v CPU). Při zavolání této funkce se spustí procesy s funkcí *cdistMultiprocessing()* s argumentem *ID*, který označuje číslo procesu. Díky argumentu *ID* každý proces pozná jakou část z databáze použít pro rozpoznání osob.

Ve výchozím nastavení je počet procesorů nastavený na 4 (spustí se 4x proces na rozpoznání osoby). Multiprocessing se spouští pouze na databázi s názvem *default*, protože u této databáze je předpoklad, že bude obsahovat obrovské množství záznamu.

startEncoding()

Provede spuštění zakódování obličejů do vektoru o velikosti 128. Obsahuje několik vstupních parametrů:

path – cesta, kde se nachází složky s fotografiemi osob. Název nebo id, které se ukládá se načte z názvu složky.

timestamp – časové razítko vytvoření.

genderRecognition – podle nastavení se spoušti také rozpoznání pohlaví či nikoliv.

selectedList – jedná se o zvolenou databázi, kde se mám zakódovány obličej s názvem (id) uložt.

Vektor obličeje uloží do zvolené databáze v argumentu *selectedList*. Všechny vytvořené zakódované obličeje, které se ukládají do uživatelem zvolené databáze se rovněž uloží do databáze s názvem *default*.

Pohlaví se rozpoznává, pokud je nahrán model do paměti GPU, který slouží pro klasifikaci pohlaví. Podle vyhodnocení pohlaví se zakódovaný obličej uloží do databáze (*maleList* – databáze mužů, *femaleList* – databáze žen).

runFacerecognition()

Tato funkce slouží pro porovnání zakódovaného obličeje neznámé osoby s databází. Obsahuje dva vstupní parametry. Prvním parametrem je *img*, na vstupu je fotka neznáme osoby. Druhým parametrem je *selectedList*, jedná se zvolenou databázi.

Při zavolání této funkce se provede nejdříve zakódování obličeje z nahrané fotky a následně se spustí samotné rozpoznávání. Pokud je spuštěný multiprocessing, tak se se volá funkce *runComparsionMultiproc()*, jinak se volá *runComparsion()*.

runComparsion()

Vstupními paramtetry jsou: *unknow_encoding* – zakódovaný obličej (vektor v Numpy poli) a *selectedList* – zvolená databáze. V případě, že se nezvolí databáze, tak se automaticky vybere databáze s názvem default, ve které jsou všechny zakódované obličeje.

Pro porovnání všech vektorů ze zvolené databáze s vektorem obličeje z nahrané fotky se používá funkce *cdist* z knihovny SciPy[12]. Pro samotný výpočet bylo provedeno vytvoření nové funkce, která ověřuje vstupní parametry před použitím:

Výpis 2.3: Výpočet Euclidovské vzdálenosti.

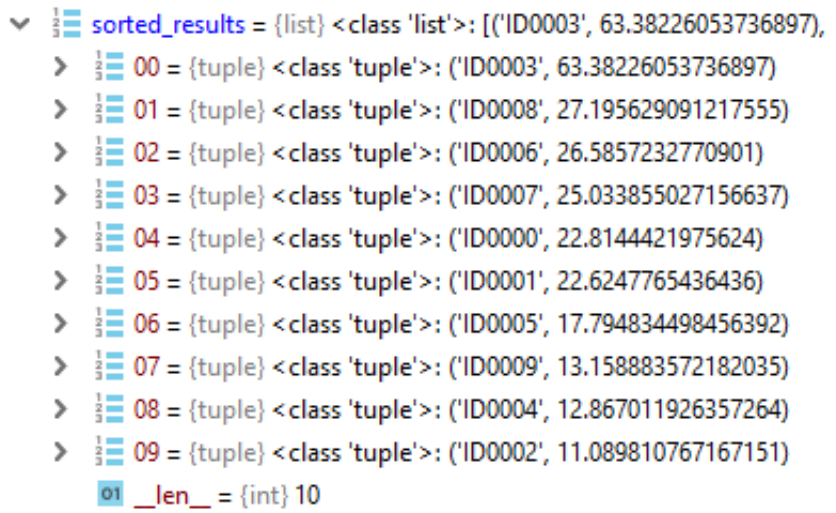
```
1 def cdistCalc(encodingList, face_to_compare):
```

[12]Open Source knihovna, která slouží pro vědecké a technické výpočty. [online] https://www.scipy.org/

```
2   if len(encodingList) == 0 or len(face_to_compare) == 0:
3     return np.empty((0))
4   return cdist(face_to_compare, encodingList)[0
```

Podmínka ošetřuje vstup před použitím prázdného seznamu nebo v případě, že by se použil prázdný vektor pro porovnání (je to nepravděpodobné, protože je to ošetřeno už při zakovávaní obličeje). Výstupem funkce cdistCalc je seznam vzdáleností mezi vektory.

Dále se provádí spojování jmen (ID) s výsledky, které vrátila funkce *cdistCalcl()*. Po spojení se provede následné seřazení a odebrání duplicitních jmen či ID ze seznamu. Funkce *runComparsion()* vrací prvních 10 nejlepších výsledků.



Obr. 2.2: Seznam výsledku před zobrazením.

runComparsionMultiproc()

Ve funkci *runComparsionMultiproc()* probíhá uložení vektoru obličeje do souborů s příponou .pickle pro procesy, které jsou spuštěné a čekají, než se soubor s vektorem vytvoří. Po uložení potom v aplikaci běží smyčka, která vyčkává na výsledek ze spuštěných procesů.

Když se uloží vektor obličeje do souboru, tak tento soubor spuštěné procesy detekuji a otevřou. V každém spuštěném procesu běží funkce *cdistMultiprocessing*. Ve spuštěných procesech už probíhá vše naprosto stejně jako ve výše popsané funkci *runComparsion*. Po provedení výpočtů vzdáleností mezi vektory, spojení a seřazení se provede uložení výsledků opět do souborů pro webovou aplikaci. Aplikace tyto soubory otevře, zpracuje a zobrazí výsledek.

2.3 Problémy a jejich řešení

Během vypracovávání bakalářské práce nastaly různé problémy, které musely být vyřešené, protože bránily k dalšímu postupu ve vypracování nebo způsobovaly neočekávané pády aplikace.

2.3.1 Problém s uložením do souborů

Aby šlo databáze znovu načíst po vypnutí aplikace bylo potřeba seznamy uložit do nějakého souboru. Nejvhodnějším a jednoduchým řešením bylo použití modulu Pickle. Pro každou databázi se vytváří samostatný soubor s daty, ale problém nastane při velkém množství záznamu.

Modul Pickle neumožnuje uložit objekt větší než 4GB, proto při velkém množství záznamu (cca 3 miliony) aplikace spadla. Tento problém bylo potřeba vyřešit, protože se do databáze *default* ukládají všechny záznamy z jiných databází a to znamená, že se může objevit v takové databáze i více než 3 miliony záznamů.

Problém byl následně vyřešen rozdělením databáze na x částí a každá část byla uložena do samostatného souboru, který se po spuštění opět načte. Soubory se nacházejí ve složce, která se jmenuje stejně jako databáze a jsou pojmenovány následovně: *název_databáze_X.pickle* (X značí číslo od 0 do 11).

2.3.2 Problém při zrychlení rozpoznávání

Už při prvotním návrhu bylo zřejmé, že bude potřeba při rozpoznávání osoby celý algoritmus nějakým způsobem zrychlit, protože nebude v databázi uloženo pár stovek záznamu, ale bude jich statisíce či dokonce miliony. Nejvhodnějším řešením bylo použití multiprocessingu, ale ještě před samotnou implementaci se provádělo několik zkušebních implementací jiných nástrojů.

Použití Numba

Numba[13] slouží pro překlad kódu, tento kód překládá za běhu aplikace (jedná se tedy o překladač JIT neboli Just In Time). To znamená, že tento nástroj měl přeložit za běhu kód pro výpočet vzdálenosti mezi vektory do strojového kódu pro GPU a výpočet měl probíhat na grafické kartě. Ovšem hned po prvních zkouškách se ukázalo, že tento nástroj nelze použít.

[13] `http://numba.pydata.org/`

CuPy a zrychlení výpočtů vzdáleností

Dalším nástrojem, který měl urychlit výpočet euklidovské vzdáleností při vysokém počtu vektoru je nástroj CuPy[14].

CuPy je open source nástroj na akceleraci maticových výpočtů na grafické kartě Nvidia (s použitím Cuda). CuPy obsahuje stejné funkce jako Numpy s tím rozdílem, že jsou agregovány na grafické kartě.

Použití tohoto nástroje se ukázalo jako správně řešení, ale pouze do chvíle než se potřebovalo použít výpočet na větším počtu vektorů. Protože převod vektorů do *cupy.ndarray* trval příliš dlouho.

V následující tabulce 2.1 je znázorněný čas potřebný pro vypočet vzdáleností. Test proběhl na počtu 998 163 vektorech obličejů. Měření času výpočtů vzdáleností bylo provedeno u CuPy, Numpy a SciPy.

Tab. 2.1: Čas výpočtů Euklidovské vzdálenosti.

	CuPy	Numpy	SciPy
Převod [s]	81,1222	-	-
Čas výpočtů [s]	0,1177	0,9714	0,1226

Převod v tabulce značí čas potřebný pro převod Numpy pole do CuPy pole. Jak je z tabulky patrné, tak převod trval 81 sekund. Výpočet vzdálenosti pomoci Numpy trvalo 971 ms na rozdíl od CuPy a SciPy, kde tento čas je 8x menší. Mezi CuPy a SciPy je rozdíl pouze 4,9 ms, proto bylo rozhodnuto, že pro výpočet Euklidovské vzdálenosti mezi vektory se použije SciPy (funkce cdist).

Problém s využitím více procesorů

Při implementaci multiprocessingu se vyskytnul problém při vytváření nových procesů. V počátku bylo navrhnuto, že se databáze rozdělí v hlavní aplikaci, a pak se rozdělená databáze pošle do každého procesu při jeho spouštěním. Procesy se měly spouštět po zavolání funkce *runComparsion()* a po dokončení úlohy se ukončit. Při malém počtu vektorů to fungovalo, ale při narůstajícím počtu se celkový čas potřebný pro inicializaci procesů prodlužoval.

Bylo to způsobeno tím, že se velký počet záznamu (stovky tisíc vektorů) muselo při každém volání *runComparsion()* rozdělit podle počtu procesů a rozdělené části poslat danému procesu (každý proces si alokuje svůj paměťový prostor a sním nadále pracuje).

Nově navrhnuté řešení, že se procesy spustí, načtou si svou část databáze a budou čekat na vytvořený soubor s jedním vektorem, bylo oproti předchozímu postupu

[14] `https://cupy.chainer.org/`

velice rychlé a jednoduché na implementaci. Nevýhodou tohoto řešení je, že určitý čas zabere ukládání a načtení souborů. Proto se pro malou databázi nevyplatí využívat více procesorového rozpoznávání. Více procesorové rozpoznávání se nespustí, pokud databáze obsahuje méně než 100k vektorů.

2.4 Představení aplikace

V rámci této bakalářské práce byla vytvořená webová aplikace na rozpoznávání osob, která umožňuje vytvářet databáze a do nich následně nahrávat fotografie obličejů.

Pro spuštění webové aplikace je podmínkou mít nainstalované všechny potřebné moduly. Pro spuštění stačí zadat v příkazové řádce *python start_web_application.py*, ihned po spuštění se začnou načítat důležité nastavení a databáze do RAM paměti. Jakmile se všechno potřebné načte, je aplikace připravená k použití.

Na web se lze dostat pomocí IP adresy stroje, kde se nachází spuštěna aplikace. Port aplikace je nastavený na 5000, lze jej libovolně měnit. Úvodní obrazovka zobrazuje sekce webu. Web obsahuje tyto stránky:

- Rozpoznáni osoby
- Nahrát data
- Nahrát foto
- Smazat
- Nastavení

Rozpoznání osoby

Tato stránka slouží k nahrání fotografie s osobou, kterou chceme vyhledat v databází. Před samotným nahráním je potřeba vybrat databázi, ve které se má osoba vyhledávat. Výsledek se objeví ve stejnojmenném sloupci. Ve sloupci se zobrazí 10 výsledků se jménem či ID, seřazené podle přesnosti v procentech. Výsledek, který je nad 50%, lze považovat za shodu.

Na obrázku A.1 je znázorněný výsledek po nahrání fotografie. Jakmile se nahraje fotografie, tak se spustí algoritmus na rozpoznávání osob. Ve sloupci Výsledek jsou seřazeny výsledky rozpoznávaní podle nejlepší shody.

Nahrát data

Na této stránce se provádí nahrávání osob do databází. Lze nahrávat soubory ve formátu zip nebo pickle. Pokud jsou fotky uloženy v archivu, tak musí být dodržena struktura (znázorněna na další straně). Pokud není dodržena struktura složek, tak se tyto složky ignorují.

Struktura

```
/ ..... kořenový adresář archivu
├── ID_1 ..... složka: osoba 1
│   ├── foto0001.jpg
│   ├── foto0002.jpg
│   └── foto0003.jpg
├── ID_2 ..... složka: osoba 2
│   └── foto0001.jpg
├── ...
└── ID_x ..... složka: osoba x
    ├── foto0001.jpg
    ├── foto0002.jpg
    ├── ...
    └── fotoxxxx.jpg
```

V případě, že je vybraná možnost nahrávání osob pomoci Pickle, tak soubor musí obsahovat dva objekty. Prvním objektem je seznam jmen (nebo ID) a ve druhém objektu se musí nacházet už vytvořené vektory obličejů.

Obrázek A.2 znázorňuje zobrazení struktury archivu před vytvořením zakódovaných obličejů (zobrazí se pouze prvních 50 osob). Pak už stačí vybrat databázi, do které se mají vektory uložit a stisknout Encoding.

Nahrát foto

Nahrát foto (na obrázku A.3) slouží pro nahrání pouze jedné fotografie obličeje do databáze. Před nahráním je potřeba vybrat databázi, do které se má vektor obličeje uložit a napsat jméno osoby. Pokud se jméno nezmění a nechá se tam předvyplněné (Jméno), tak se použije název souboru jako jméno osoby.

Smazat

Slouží ke smazání osoby z databáze. K úspěšnému smazání dané osoby je potřeba vybrat databází a napsat přesně jméno nebo ID, které se nachází v databází. Před smazáním je uživatel dotázán, zdali opravdu chce smazat zvolenou osobu z databáze. Když v dialogovém okně klikne na Ano, provede se prohledání všech záznamu. Pokud je nalezena shoda, tak se všechny vektory z databáze smažou. Ve sloupci Výsledek se objeví počet smazaných vektorů. Na obrázku A.4 je zobrazena stránka po smazání osoby z databáze.

Nastavení

Tato položka menu obsahuje dvě stránky, první stránkou je Globální nastavení a druhou je Uložit.

- **Globální:** Obsahuje veškeré nastavení aplikace, které lze za běhu aplikace měnit (obrázek A.5). Nachází se zde přidání a odebrání databáze, spuštění/zastavení více procesorového rozpoznávání a nastavení poštu procesů, aktivace/deaktivace rozpoznávání pohlaví a nastavení prahu přesnosti.
- **Uložit:** Slouží pro uložení všech databází do souborů s příponou pickle, tyto soubory se následně používají pro opětovné nahrání databází po spuštění aplikace. Po uložení všech databází je uživatel informován o počtu databází a celkovém počtu záznamu v nich.

3 Výsledky

V této kapitole jsou představeny výsledky měření rychlosti a přesnosti frameworků OpenFace a Face Recognition. Podkapitola 3.2 obsahuje měření rychlosti rozpoznávání vytvořené webové aplikace.

3.1 Výsledky a hodnocení frameworků

Při porovnávání programu OpenFace a Face Recognition byla vytvořena databáze obrázků veřejně dostupných obrázků s volnou licencí. Jedná se snímky osob vyfocených z různých úhlu, vzdáleností a s různým pozadím. Byly vybrány snímky čtyř známých osob kvůli dobré dostupnosti fotografií na internetu. Testovací měření probíhalo pouze na procesoru Inter Core i7-4790K.

3.1.1 Průměrné časy podle velikosti

Tato kapitola představuje, jak se měnil čas s velikostí obrázků. Měření proběhlo na barevných a černobílých snímcích.

Barevné obrázky (RGB)

V první řadě byla vytvořena složka s barevnými obrázky o velikosti 3264x2448 pixelů. V této složce bylo 30 obrázků, aby se vyzkoušelo, jak jsou výpočetní časy závislé na velikosti obrázků. Pro každé nové měření průměrného času byly obrázky zmenšeny o polovinu. Průměrné časy lokalizace obličeje si lze prohlédnout v tabulce 3.1 a průměrné časy zakódování v tabulce 3.2

Tab. 3.1: Průměrné časy lokalizace obličeje na RGB snímcích

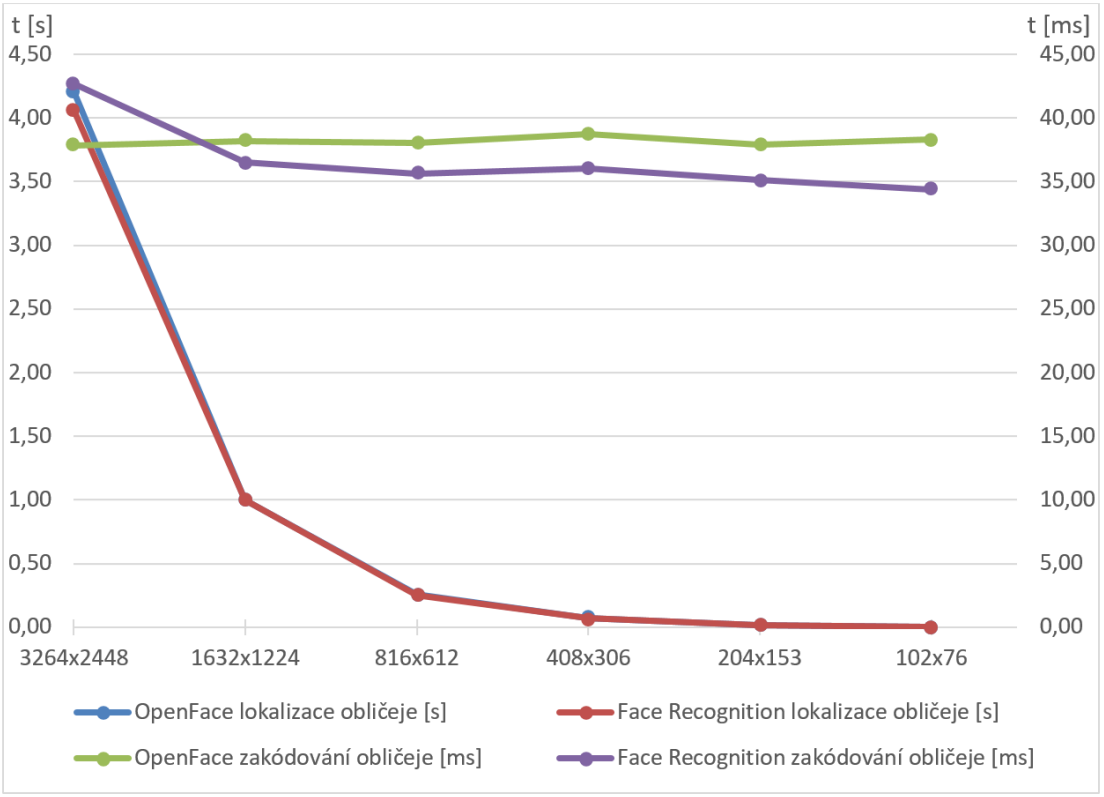
	OpenFace [s]			Face Recognition [s]		
Velikost obr.	Minimální	Maximální	Průměr	Minimální	Maximální	Průměr
3264x2448	3,97132	4,20855	3,99686	3,95589	5,018	4,06072
1632x1224	0,99038	1,0026	0,99588	0,99343	1,00586	0,99935
816x612	0,25129	0,25771	0,25316	0,25046	0,25538	0,25282
408x306	0,06575	0,07364	0,06681	0,06541	0,07125	0,06802
204x153	0,01668	0,01882	0,01713	0,01673	0,01903	0,01725
102x76	0,00403	0,00481	0,00415	0,00395	0,00409	0,004

Ze změřených časů je zřejmé, že lokalizace obličeje je u OpenFace a Face Recognition téměř stejná. Je to způsobeno tím, že oba dva programy používají k lokalizaci

Tab. 3.2: Průměrné časy zakódování, RGB.

	OpenFace [s]			Face Recognition [s]		
Velikost obr.	Minimální	Maximální	Průměr	Minimální	Maximální	Průměr
3264x2448	0,03727	0,0391	0,03787	0,0343	0,07512	0,04269
1632x1224	0,03713	0,04549	0,03821	0,03387	0,06839	0,03649
816x612	0,03742	0,03884	0,03803	0,03403	0,06804	0,03564
408x306	0,03744	0,04456	0,03876	0,03366	0,06929	0,03603
204x153	0,03895	0,03895	0,03792	0,03375	0,03757	0,0351
102x76	0,03734	0,0401	0,0383	0,03384	0,03639	0,03439

stejnou knihovnu Dlib. Odchylky v řádech milisekund jsou pravděpodobně způsobeny procesy, které běží na pozadí a můžou způsobit zpomalení procesu testované aplikace. Průměrné časy zakódování se liší pouze v řádu jednotek milisekund a na velikosti snímku takřka nezáleží, jak je vidět na grafu 3.1



Obr. 3.1: Průměrné časy lokalizace a zakódování obličeje, RGB.

Černobílé obrázky (BW)

Stejné měření bylo provedeno u černobílých obrázků. Časy lokalizace obličeje jsou uvedeny v tabulce 3.3 a čas zakódování obličeje (tj. vytvoření vektoru) je v tabulce 3.4. Všechny obrázky, které byly použity v předchozím měřením, byly převedeny na černobílé snímky.

Tab. 3.3: Průměrné časy lokalizace obličeje na BW snímcích.

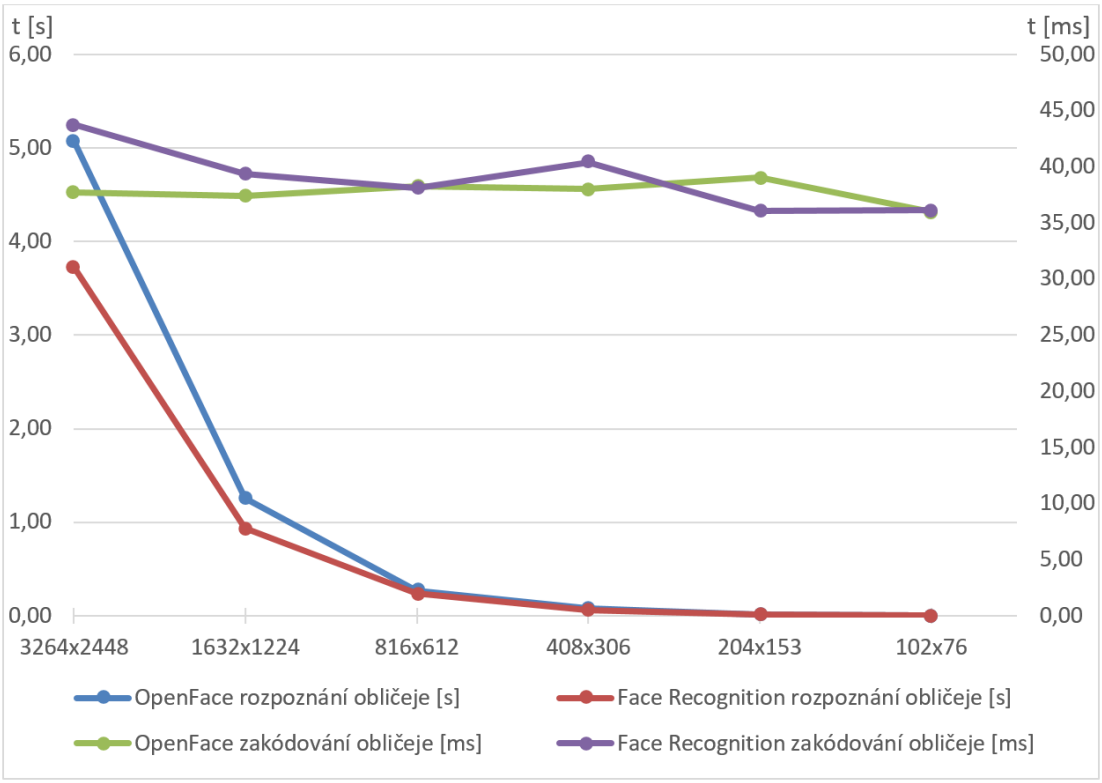
	OpenFace [s]			Face Recognition [s]		
Velikost obr.	Minimální	Maximální	Průměr	Minimální	Maximální	Průměr
3264x2448	3,97022	5,07273	4,05423	3,6585	4,04946	3,72302
1632x1224	0,99917	1,26024	1,01099	0,9156	1,20861	0,93374
816x612	0,25041	0,2721	0,25283	0,23172	0,32669	0,23548
408x306	0,06569	0,07659	0,06824	0,06158	0,08793	0,06471
204x153	0,01679	0,0171	0,01692	0,01573	0,01601	0,01586
102x76	0,00000	0,00541	0,00397	0,00364	0,00382	0,00369

Tab. 3.4: Průměrné časy zakódování, BW.

	OpenFace [s]			Face Recognition [s]		
Velikost obr.	Minimální	Maximální	Průměr	Minimální	Maximální	Průměr
3264x2448	0,03738	0,03916	0,03773	0,0366	0,07716	0,04378
1632x1224	0,03714	0,03821	0,03743	0,03618	0,07167	0,03935
816x612	0,03776	0,03901	0,03823	0,03582	0,07214	0,03813
408x306	0,03708	0,04186	0,038	0,03578	0,0748	0,04039
204x153	0,05246	0,05246	0,03906	0,03566	0,03648	0,03609
102x76	0,00000	0,04165	0,03599	0,03579	0,03658	0,03611

Jak jde na grafech 3.1 a 3.2 vidět, tak jsou mezi RGB a BW snímcích malé rozdíly, které mohou být opět způsobeny procesy na pozadí. Dá se říct, že čas lokalizace a zakódování obličeje není závislý na barvě snímku, ale závisí na velikosti snímku. V případě zakódování obličeje velikost snímku nehraje roli, protože do neuronové sítě vstupuje upravená velikost obličeje (tj. 150x150px).

Měření průměrného času bylo také provedeno u snímků, které jsou různě opravené. Například změněný jas snímku nebo rozostření snímku. Průměrné časy byly stejné jako u barevných nebo černobílých obrázků, tudíž na rychlosti lokalizace a zakódování obličeje na kvalitě snímku nezáleží.



Obr. 3.2: Průměrné časy lokalizace a zakódování obličeje, BW.

3.1.2 Přesnost

Byly porovnávány přesnosti rozpoznávání za různých podmínek, jako je změna jasu nebo rozmazání fotografie. Různě upravené snímky byly porovnávány se čtyřmi výchozími snímky. Bylo použito 30 RGB snímků o velikosti 1632x1224 px (stejné snímky jako v předchozích testech) a ty byly následně v programu IrfanView upraveny.

Rozostřené snímky

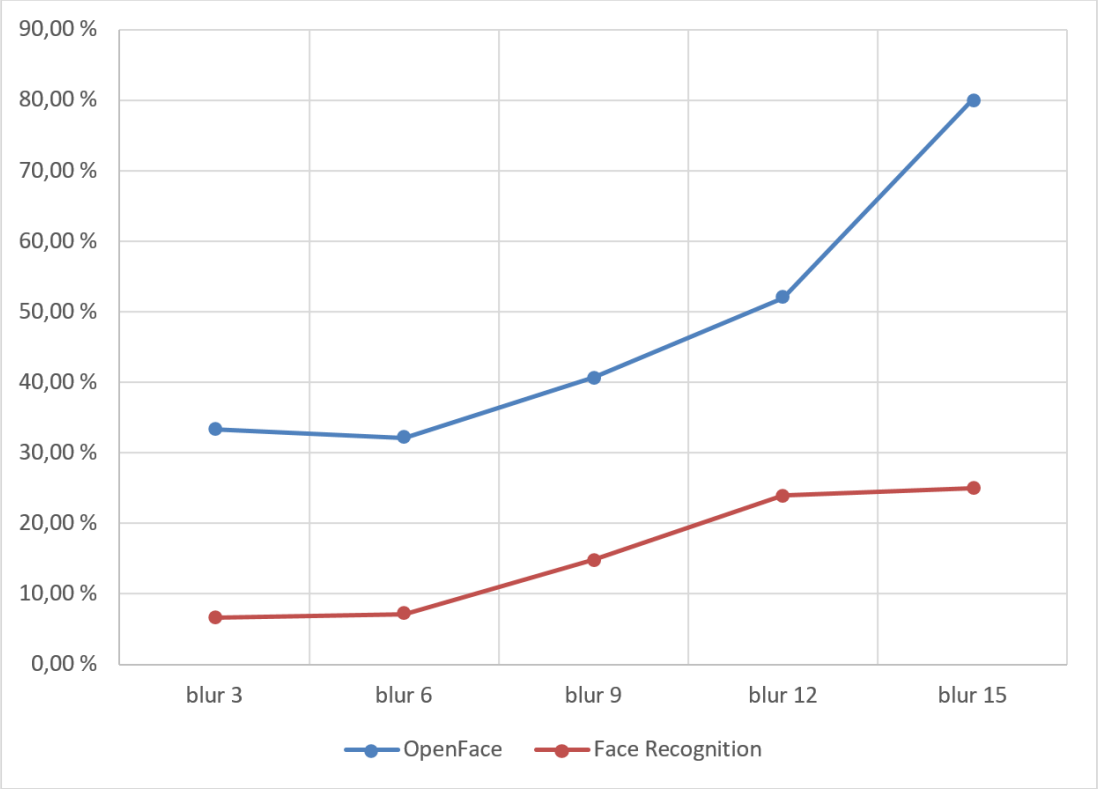
Za pomoci IrfanView a pluginu Gausian blur bylo provedeno rozostření. Zvolené rozostření je 3, 6,. . . , 15.

V tabulce 3.5 Počet detekcí udává na kolika snímcích se podařilo detekovat obličej. Chybně rozpoznáno znamená počet špatně rozpoznaných obličejů. Počet detekcí je u OpenFace a Face Recognition stejný, protože oba dva frameworky k detekci obličeje využívají knihovnu Dlib. Z tabulky je patrné, že u největšího rozmazání nebyl detekovaný obličej na deseti snímcích ze třiceti. Jak je v tabulce 3.5 a grafu 3.3 uvedeno, tak se přesnost rozpoznání zhoršovala, čím více byl snímek rozostřený.

Tab. 3.5: Chyba rozpoznání na rozostřených snímcích.

		OpenFace		Face Recognition	
Rozostření	Počet detekcí	Chybně rozpoznáno	%	Chybně rozpoznáno	%
blur 3	30	10	33,33%	2	6,67%
blur 6	28	9	32,14%	2	7,14%
blur 9	27	11	40,74%	4	14,81%
blur 12	25	13	52,00%	6	24,00%
blur 15	20	16	80,00%	5	25,00%
Průměr	26	12	47,64%	4	15,53%

Z výsledku vyplývá, že neuronová síť OpenFace si neumí si s kvalitou snímku, tak dobře poradit, jako neuronová síť knihovny Dlib, kterou používá Face Recognition.



Obr. 3.3: Chyba rozpoznání závislá na ostrosti.

Věk osoby

K ověření, zda je závislá přesnost rozpoznání na věku osoby, byla vytvořena databáze snímků osob. Z toho důvodu byly opět vyhledány snímky třech známých osob, které byly z roku 2017, 2007, 1997, 1987 (+/- 1 rok).

Jak je z tabulky 3.6 patrné, tak OpenFace má problém se správným rozpoznáním, pokud se pro rozpoznávání použijí starší snímky. Face Recognitiom chybně rozpoznal osobu ve třech případech. Framework OpenFace má průměrnou přesnost rozpoznání osoby 68% a Face Recognition má průměrnou přenost 97%.

Tab. 3.6: Chyba rozpoznání podle staří fotografie.

		OpenFace		Face Recognition	
Nazev	Počet obr.	Chybně rozpoznáno	%	Chybně rozpoznáno	%
Arnold Schwarzenegger	31	13	41,94%	3	9,68%
Dwayne Johnson	10	0	0,00%	0	0,00%
Sylvester Stallone	13	7	53,85%	0	0,00%
Průměr	18	6,6	32%	1	3%

Tabulka 3.7 představuje počet chybně provedených rozpoznání osob. Různé fotografie osoby Sylvestra Stallona (název obrázků reprezentuje rok, kdy byl snímek pořízen) byly porovnány s aktuálním snímkem (IMG1 až IMG4).

IMG1 - Arnold Schwarzenegger

IMG2 - Dwayne Johnson

IMG3 - Sylvester Stallone

IMG4 - Vin Diesel

Tab. 3.7: Ukázka přesnosti závislá na věku osoby, Sylvester Stallone.

	OpenFace				Face Recognition			
Název	IMG1	IMG2	IMG3	IMG4	IMG1	IMG2	IMG3	IMG4
1987_0.jpg	1,893	0,9654	1,0418	1,033	0,8402	0,8565	0,5009	0,6771
1987_1.jpg	1,9611	0,8498	0,7296	1,0102	0,8668	0,8829	0,4545	0,7235
1997_0.jpg	1,7059	0,6589	0,7067	1,2044	0,8685	0,8563	0,5474	0,7414
1997_1.jpg	2,2554	0,9827	0,7421	1,1522	0,9057	0,8017	0,502	0,7175
1997_2.jpg	1,9174	1,0991	0,4624	1,0859	0,9025	0,8628	0,4618	0,7434
2007_0.jpg	1,6323	1,6008	0,6723	1,3959	0,8632	0,892	0,4455	0,7541
2007_1.jpg	1,6914	1,0239	0,3405	1,119	0,8506	0,8777	0,3017	0,7397
2007_2.jpg	1,4816	1,2802	0,6289	0,7107	0,8098	0,8789	0,4457	0,7692
2007_3.jpg	1,6413	1,1554	0,4225	1,1945	0,8022	0,8672	0,4172	0,7385
2007_4.jpg	1,6989	1,2159	0,9426	1,3558	0,8519	0,8881	0,463	0,7304
2017_0.jpg	1,7228	1,1076	0,5285	1,2308	0,8864	0,793	0,4855	0,6824
2017_1.JPG	1,468	1,6846	0,7493	1,0434	0,8623	0,9020	0,4477	0,7274
2017_2.jpg	1,5371	1,4558	0,7195	0,9050	0,8755	0,8536	0,3934	0,7014

Vyhodnocení, zda se jedná o chybu rozpoznání či nikoliv bylo prováděno na základě porovnání vzdálenosti. To znamená, že vypočítaná vzdálenost u OpenFace menší než 1 znamená shodu osoby. U Face Recognition musí být vzdálenost menší

než 0,6, aby se jednalo o shodu. Cokoliv nad zvolenou hranici lze považovat, že se dvě osoby neshodují.

Pokud byla vypočítaná vzdálenost u dvou různých osob přesto menší než zvolená hranice (tudíž mělo by se jednat o shodu), tak tento výsledek lze považovat za chybné rozpoznání a naopak. V tabulce 3.7 jsou takové chybné rozpoznáni zvýrazněné červeně, OpenFace provedl špatné rozpoznání v sedmi případech, zatímco Face Recognition neudělal chybu.

Změna světlosti snímku

Třiceti snímkům o velikosti 1632 x 1224 pixelů byl změněný jas v programu IrfanView na hodnoty: -225, -200, -150 (pro ztmavení snímku) a 150, 200, 225 (pro zesvětlení).

Tabulka 3.8 popisuje, jak moc je rozpoznání osoby závisle na světlosti fotografie. Je patrné, že při velmi tmavém snímku nebo naopak velmi světlém snímku má OpenFace vysoké procento chyby rozpoznání na rozdíl od Face Recognition.

Chybné rozpoznání obličeje je způsobeno například, když tmavý obličej je na tmavém pozadí nebo naopak. Jakmile je pozadí a barva obličeje podobná, tak programy mají problém s detekci obličeje a s následným rozpoznáním.

Tab. 3.8: Chyba rozpoznání při změně světlosti snímku.

		OpenFace		Face Recognition	
Jas obr.	Počet detekcí	Chybně rozpoznáno	%	Chybně rozpoznáno	%
světlost -225	26/30	21	80,77 %	2	7,69 %
světlost -200	29/30	15	51,72 %	1	3,45 %
světlost -150	30/30	7	23,33 %	1	3,33 %
světlost 150	30/30	14	46,67 %	1	3,33 %
světlost 200	28/30	22	78,57 %	2	7,14 %
světlost 225	24/30	23	95,83 %	5	20,83 %
Průměr	28/30	17	63%	2	8%

Skrytí obličeje slunečními brýlemi

Snímky osob (36 snímků), které mají na obličeji nasazené sluneční brýle byly porovnány se stejnými osobami, ale bez slunečních brýlí. Osoby byly na snímku v různém úhlu natočení. Na některých snímcích šla rozeznat poloha očí, protože brýle nebyly dostatečně tmavé. Vesměs většina snímku obsahovala obličej s brýlemi, kde byly očí (někde i obočí) úplně skryty.

Výsledek testu rozpoznání osob se slunečními brýlemi je v tabulce 3.9. Stejně jako v předchozích testech opět OpenFace chybně rozpoznal osobu a to v 55,88%

případů (přesnost rozpoznání 44,12%). Face Recognition udělal chybu jen ve třech (8,82%) případech, přesnost frameworku Face Rocognition je 91,18%.

Tab. 3.9: Chyba rozpoznání při skrytí obličeje slunečními brýlemi.

	OpenFace		Face Recognition	
Počet detekcí	Chybně rozpoznáno	%	Chybně rozpoznáno	%
34/36	19	55,88%	3	8,82%

Kombinace snímků při rozpoznávání

Jelikož v předchozích testech byly vytvářený složky, ve kterých byly různě upravené snímky, proto jako poslední test byla provedla kombinace všech snímků. To znamená, pokud se nacházelo ve složce 30 snímků, tak se provedla kombinace těchto snímků mezi sebou. V tabulce 3.10 je uvedeno, kolik složek bylo použito při testu, kolik průměrně bylo ve složce obrázků a kolik kombinaci v každé složce bylo průměrně provedeno. Výsledek chybného rozpoznání je uvedený pouze v procentech. OpenFace provedl chybné rozpoznání ve 27% (tj. přesnost 73%) a Face Recognition pouze ve 4% (přenost 96%).

Tab. 3.10: Chyba rozpoznání s kombinaci snímků.

Celkem		Průměr			
Složek	Obr.	Obr. ve složce	Kombinací	OpenFace chyba	Face Recognition chyba
63	1623	22	265	27%	4%

3.1.3 Hodnocení frameworků

Při posledním testu rozpoznávání osob s kombinací snímků vyšlo najevo, že OpenFace oproti Face Recognition má horší rozpoznávání osob. To bylo patrné už při prvních testech přesností, v posledním testu se to jen potvrdilo. Značné rozdíly v chybném rozpoznávání byly zaznamenány především při úpravě snímků (rozostření, změna jasu).

OpenFace také chybně rozpoznával osoby, které mají na snímcích různý věk. Problém stárnutí lidí nelze eliminovat, ale pouze k tomu přizpůsobit algoritmy. Face Recognition se mnohem lépe s problémem staří fotografie poradil.

Z výsledků měření lze usoudit, že nejvhodnější program k rozpoznávání osob je Face Recognition. Vesměs ve všech testech průměrná chyba rozpoznání Face Recognition nepřesáhla 10%, kromě testu rozostření, kde byla průměrná chyba 15,53% oproti OpenFace, kde průměrná chyba byla 47,64%.

3.2 Webová aplikace: rychlost rozpoznávání

Čas se měřil nejdříve při rozpoznávání klasickým způsobem (použilo se pouze jedno jádro), pak se měřil čas rozpoznávání při použití více jáder.

Pro měření času rozpoznání závislého na velikosti databáze bylo vybráno 9 snímků o velikosti 500x500 px. Pro každý snímek se měřil čas rozpoznání aby se vyloučilo např. zatížení CPU jiným procesem následně bylo všech 9 času zprůměrováno.

Měření času proběhlo na počítači s těmito parametry: CPU: Intel Core i5-8400 (2.8 GHz, 6 fyzických jáder), RAM: 16GB

Protože systém byl limitovaný nedostatečně velkou RAM paměti, tak měření času bylo provedeno na maximálním počtu záznamu, kterého šlo dosáhnout (tj. 10,3 milionů vektorů).

V tabulce 3.2 jsou uvedeny časy výpočtu vzdálenosti a celkový čas. Celkový čas představuje čas měřený od nahrání fotografie do aplikace, až po zobrazení výsledku rozpoznávání. Počet procesů odpovídá počtu využitých jáder CPU. Rychlost rozpoznávání závisí na výkonu počítače, tím pádem při použití výkonnějšího procesoru lze docílit rychlejšího rozpoznání osoby.

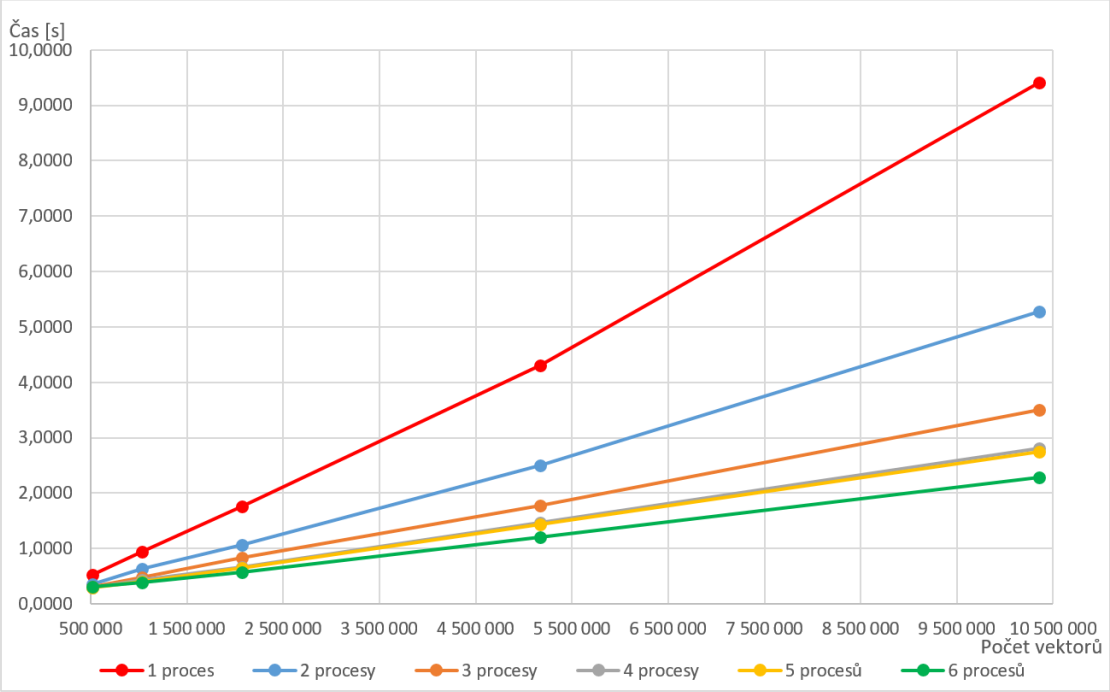
Čas rozpoznání při použití čtyř nebo pěti procesů je téměř stejný. Je to způsobeno rozdělením ukládaných databází do 12 souborů, v případě použití 5 procesů si první tři procesy načtou po dvou souborech, ale poslední dva procesy musí načíst zbytek (tj. celkem 6 souborů). V případě použití čtyř procesů, každý proces načítá tři soubory (počet načtených souborů je ve všech procesech stejný). Hlavní aplikace po spuštění rozpoznávání čeká na výsledky od všech procesů, tím pádem čas celkového rozpoznání závisí na doběhnutí procesu s nejvyšším počtem načtených souborů (v případě pěti procesů to jsou 3 soubory).

Aby se zabránilo nerovnoměrnému rozdělení souborů do procesů, je potřeba vhodně zvolit počet procesů, například počet procesů určit společným dělitelem. V případě 6 jádrového CPU je nejvyšším společným dělitelem číslo 6, tudíž lze použít všechny jádra k rozpoznávání.

Tab. 3.11: Čas rozpoznání osoby podle počtu procesorů.

	Celkový čas [s]					
Počet záznamů	1 proces	2 procesy	3 procesy	4 procesy	5 procesů	6 procesů
517 566	0,5249	0,3601	0,3092	0,2952	0,2954	0,3097
1 035 132	0,9369	0,6289	0,4756	0,4261	0,3954	0,3875
2 070 264	1,7640	1,0676	0,8343	0,6697	0,6408	0,5678
5 172 660	4,3021	2,5011	1,7802	1,4649	1,4287	1,1982
10 351 320	9,4119	5,2771	3,4943	2,8039	2,7380	2,2780

Pokud se v databází nachází více než jeden milion záznamů, tak rozpoznávání je nejrychlejší při použití 6 procesů (viz. tabulka 3.2). V případě databáze o velikosti 10,3 milionů záznamu (vektorů) trvalo rozpoznání osoby průměrně 9,4119 sekund při použití pouze jednoho procesu. Když se použilo 6 procesů, tak se čas rozpoznávání zlepšil na 2,278 sekund. Graf 3.4 zobrazuje závislost času rozpoznávání osoby na počtu vektorů v databází.



Obr. 3.4: Graf závislosti času rozpoznání na počtu vektorů v databázi.

4 Závěr

Cílem této bakalářské práce bylo prostudovat implementace vycházející z projektu FaceNet, respektive frameworky OpenFace a Face Recognition. Implementace se měly porovnat z hlediska výpočetních času a přesnosti. Následně vytvořit webovou aplikaci na rozpoznávání osob, která umožní rozpoznání osoby.

První kapitola se zabývá teoretickou částí problematiky. Druhá kapitola je zaměřená na praktické testování implementací OpenFace a Face Recognition (tj. přesnost a výpočetní čas). V kapitole 2.2 je návrh webové aplikace na rozpoznávání osob. Tato kapitola obsahuje návrh algoritmu pro zrychlení rozpoznávání osob a problémy, které se musely vyřešit. Kapitola 3 představuje výsledky řešené práce.

V rámci bakalářské práce byly zjištěny výpočetní časy při různé velikosti obrázků. Byly porovnávány barevné i černobílé snímky. Největší velikost obrázků byla zvolena 3264x2448px a následně byly obrázky zmenšeny (vždy o polovinu), až na velikost 102x76 pixelů. Při měření se zjistilo, že se čas lokalizace obličeje u OpenFace a Face Recognition liší pouze o pár milisekund. Dále se měřila přesnost při zhoršené kvalitě snímku, snížené a zvýšené světlosti. Součástí měření byl test přesnosti závislý na stáří fotografie. Taktéž byla měřena přesnost rozpoznání, pokud má osoba nasazeny sluneční brýle. Výsledky testů ukázaly, že OpenFace je ve všech testech horší než Face Recognition. Přesnost při nasazených slunečních brýlí byla u OpenFace pouze 44,12% a u frameworku Face Recognition byla přesnost 91,18%. Měřením přesnosti rozpoznávání na kombinací snímků byla zjištěna přesnost OpenFace 73% a Face Recognition 96%.

Hlavním přínosem bakalářské práce je webová aplikace na rozpoznávání osob, která je schopná rozpoznat 10 milionu osob za 9,4 sekundy a při použití více procesorové funkce to zvládne za 2,2 sekundy (v případě použití šesti jáder procesorů). Uživatel má možnost vybrat databázi, na které se bude provádět rozpoznávání osob. Aplikace umožňuje použití automatického rozdělení osob do databáze mužů a žen, takovéto rozdělení může výrazně snížit čas rozpoznávání. Rozdělení osob do různých podmnožin je možné provést také ručně.

Navrženou webovou aplikaci lze použít ve všech situacích, kdy je potřeba identifikace osob. Své využití by dokázala najít např. ke kontrole vstupu oprávněných osob do různých objektů, na letištích při celních kontrolách nebo při pátrání po konkrétních osobách.

Literatura

[1] Wenyi Zhao, Rama Chellappa, P Jonathon Phillips, and Azriel Rosenfeld. Face recognition: A literature survey. *ACM computing surveys (CSUR)*, 35(4):399–458, 2003.

[2] Dominik Scherer, Hannes Schulz, and Sven Behnke. Accelerating large-scale convolutional neural networks with parallel graphics multiprocessors. pages 82–91, 09 2010. doi:10.1007/978-3-642-15825-4_9.

[3] Understanding convolutional neural networks for nlp. URL: http://www.wildml.com/2015/11/understanding-convolutional-neural-networks-for-nlp/.

[4] Brandon Amos, Bartosz Ludwiczuk, Mahadev Satyanarayanan, et al. Openface: A general-purpose face recognition library with mobile applications. *CMU School of Computer Science*, 6, 2016.

[5] Kevin W Bowyer. Face recognition technology: security versus privacy. *IEEE Technology and society magazine*, 23(1):9–19, 2004.

[6] Madhavi R Bichwe, SR Bichwe, and Sugandha Satija. Design and implementation of resampling techniques for face recognition using classical lda algorithm in matlab. *International Journal of Computer Applications*, 152(6), 2016.

[7] Divyarajsinh N Parmar and Brijesh B Mehta. Face recognition methods & applications. *arXiv preprint arXiv:1403.0485*, 2014.

[8] Matthew A Turk and Alex P Pentland. Face recognition using eigenfaces. In *Proceedings. 1991 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 586–591. IEEE, 1991.

[9] Haoxiang Li, Zhe Lin, Xiaohui Shen, Jonathan Brandt, and Gang Hua. A convolutional neural network cascade for face detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5325–5334, 2015.

[10] Patrick J Grother. Face recognition vendor test 2002 supplemental report| nist. Technical report, 2004.

[11] Lacey Best-Rowden and Anil K Jain. Longitudinal study of automatic face recognition. *IEEE transactions on pattern analysis and machine intelligence*, 40(1):148–162, 2018.

[12] Navneet Dalal and Bill Triggs. Histograms of oriented gradients for human detection. In *international Conference on computer vision & Pattern Recognition (CVPR'05)*, volume 1, pages 886–893. IEEE Computer Society, 2005.

[13] Daniel Maturana and Sebastian Scherer. Voxnet: A 3d convolutional neural network for real-time object recognition. In *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 922–928. IEEE, 2015.

[14] Jiazheng Shi, Ashok Samal, and David Marx. How effective are landmarks and their geometry for face recognition? *Computer vision and image understanding*, 102(2):117–133, 2006.

[15] Eva Volná. Evoluční algoritmy a neuronové sítě. *Ostrava: Ostravská univerzita v Ostravě. Dostupné z: http://physics. ujep. cz/˜ mmaly/vyuka-/MPVT_II/Heuristiky/Evolucni_algoritmy_a_neuronove _siteSOMA+ Genetika. pdf*, 2012.

[16] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *nature*, 521(7553):436, 2015.

[17] LeCun Yann. Lenet-5, convolutional neural networks, 2013.

[18] Andrej Karpathy et al. Cs231n convolutional neural networks for visual recognition. *Neural networks*, 1, 2016. URL: `http://cs231n.github.io/convolutional-networks/`.

[19] Yaniv Taigman, Ming Yang, Marc'Aurelio Ranzato, and Lior Wolf. Deepface: Closing the gap to human-level performance in face verification. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1701–1708, 2014.

[20] Erik Learned-Miller, Gary B Huang, Aruni RoyChowdhury, Haoxiang Li, and Gang Hua. Labeled faces in the wild: A survey. In *Advances in face detection and facial image analysis*, pages 189–248. Springer, 2016.

[21] Florian Schroff, Dmitry Kalenichenko, and James Philbin. Facenet: A unified embedding for face recognition and clustering. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 815–823, 2015.

[22] Lior Wolf, Tal Hassner, and Itay Maoz. *Face recognition in unconstrained videos with matched background similarity.* IEEE, 2011.

[23] Davis E King. Dlib-ml: A machine learning toolkit. *Journal of Machine Learning Research*, 10(Jul):1755–1758, 2009.

[24] Dong Yi, Zhen Lei, Shengcai Liao, and Stan Z Li. Learning face representation from scratch. *arXiv preprint arXiv:1411.7923*, 2014.

[25] Adam Geitgey and J Nazario. Face recognition. URL: `https://github.com/ageitgey/face_recognition`.

[26] Open source initiative. URL: `https://opensource.org/licenses/`.

[27] Numpy. URL: `http://www.numpy.org/`.

[28] Octavio Arriaga, Matias Valdenegro-Toro, and Paul Plöger. Real-time convolutional neural networks for emotion and gender classification. *arXiv preprint arXiv:1710.07557*, 2017.

[29] Fei Wang, Liren Chen, Cheng Li, Shiyao Huang, Yanjie Chen, Chen Qian, and Chen Change Loy. The devil of face recognition is in the noise. *arXiv preprint arXiv:1807.11649*, 2018.

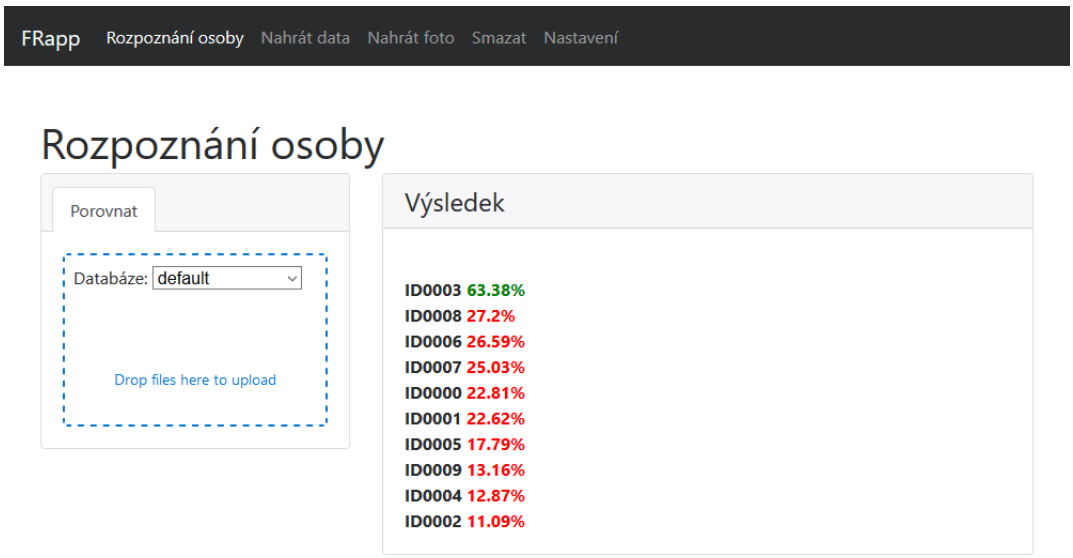
[30] Numpy. URL: `https://docs.python.org/`.

Seznam symbolů, veličin a zkratek

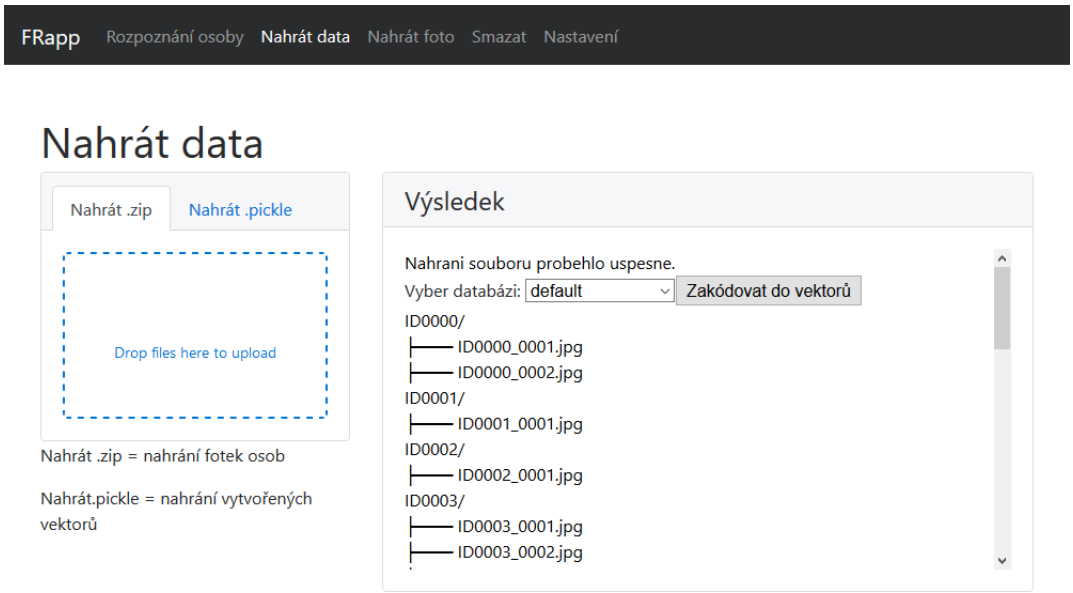
FRVT Face Recognition Vendor Test
BSD Berkeley Software Distrobution
BW Blacl, White
RGB Red, Green, Blue
CNN Convolutional neural network
HOG Histogram of Oriented Gradients
GPU Grafická procesorová jednotka
CPU Centrální procesorová jednotka
LFW Labeled Faces in the Wild
JIT Just In Time
px Pixel
SQL Structured Query Language

Seznam příloh

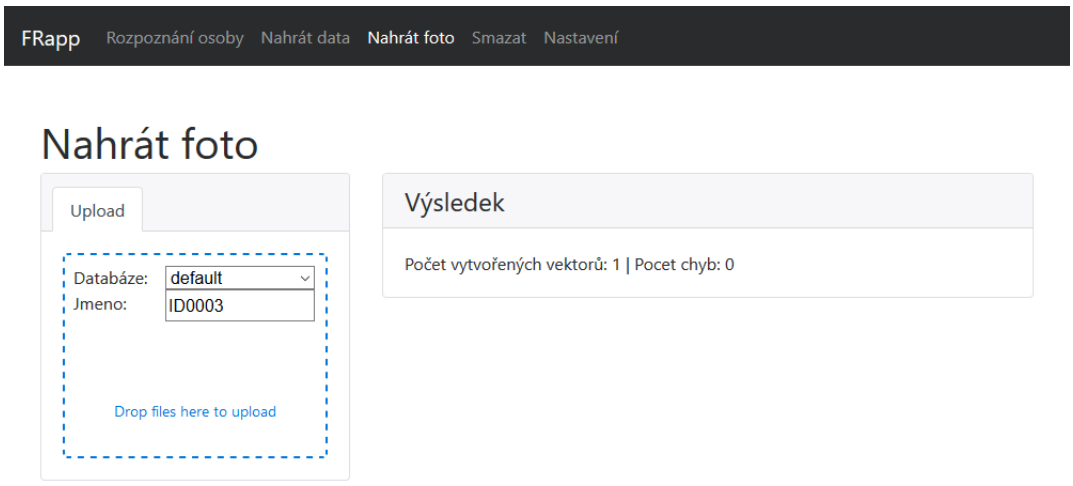
A Ukázka webové aplikace



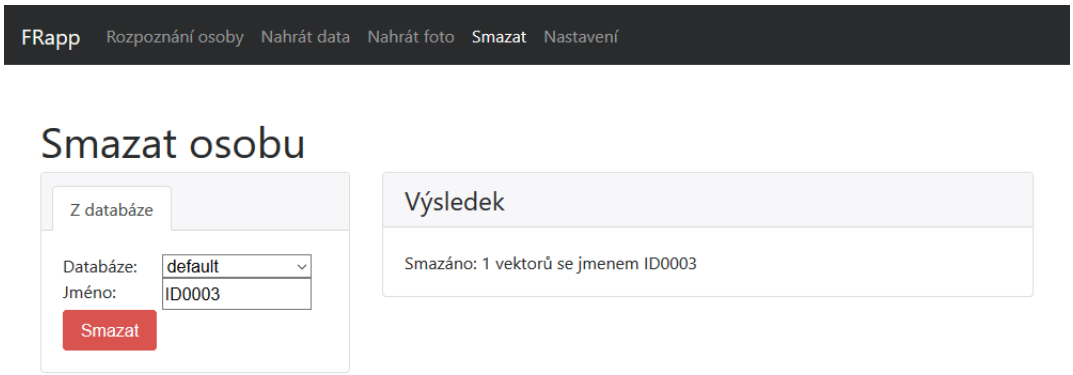
Obr. A.1: Výsledek rozpoznání osoby.



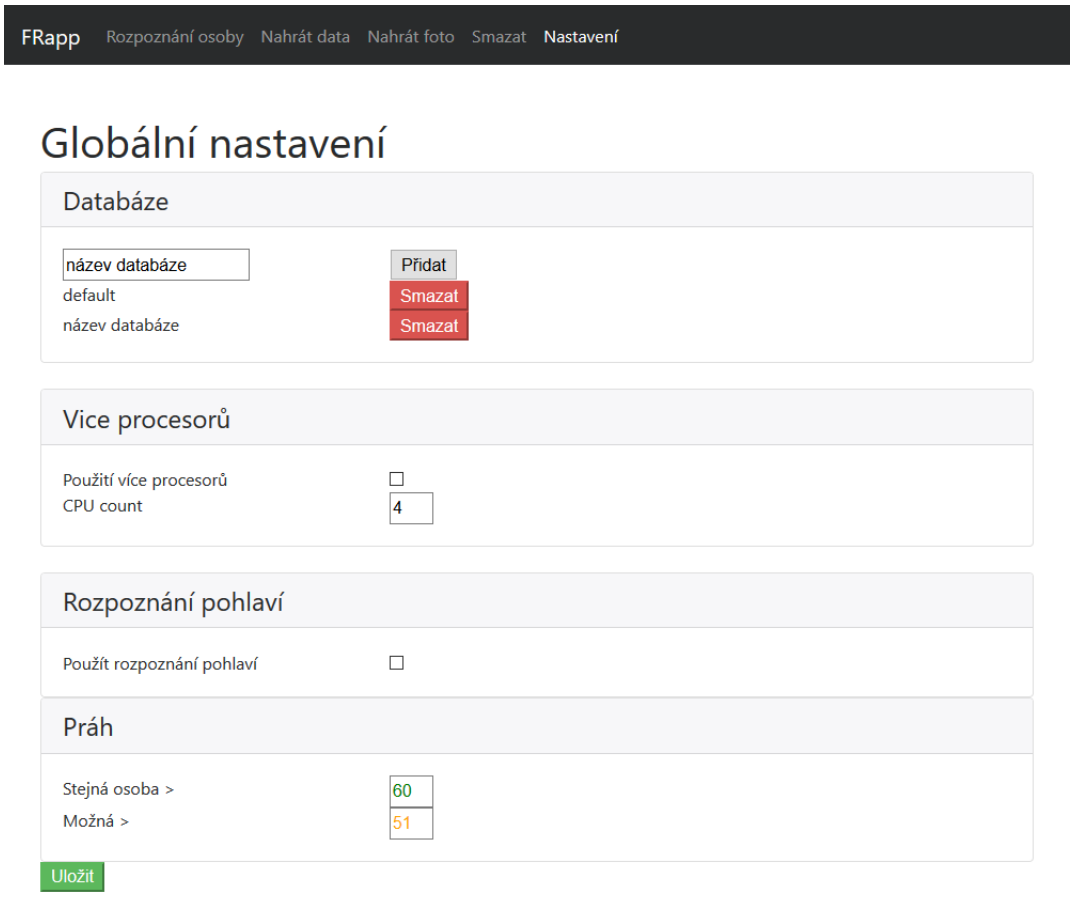
Obr. A.2: Nahraní dat do databáze.



Obr. A.3: Nahraní jedné fotografie do databáze.



Obr. A.4: Smazání osoby z databáze.



Obr. A.5: Nastavení webové aplikace.

B Obsah přiloženého CD

Přiložené CD obsahuje elektronickou verzi bakalářské práce ve formátu PDF a zdrojové kódy vytvořené webové aplikace. Po rozbalení archivu *FRapp.zip* je potřeba před spuštěním stáhnout a nainstalovat všechno potřebné knihovny. Seznam knihoven se nachází v souboru *requirements.txt*, pro stažení a nainstalování celého seznamu stačí použít příkaz *pip install requirements.txt*. Následné spuštění webové aplikace proběhne příkazem *python start_web_application.py*.

- `/` ..... kořenový adresář
  - `dokumentace` ..... elektronická verze bakalářské práce
    - `xlindo03.pdf`
  - `zdrojove_kody` ..... zdrojové kódy webové aplikace
    - `FRapp.zip` ..... zdrojové kódy